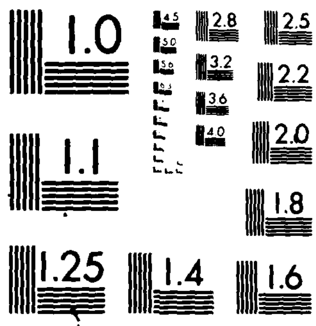
AD-A098 647 NAVAL UNDERWATER SYSTEMS CENTER NEW LONDON CT NEW LO--ETC F/G 9/2
MACSYMA: A PROGRAM FOR COMPUTER ALGEBRAIC MANIPULATION (DEMONST--ETC(U)
MAR 81 R D DRINKARD, N K SULINSKI
UNCLASSIFIED NUSC-TD-6401 NL

END
DATE
FILMED
6-81
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963-A

AD A098647

AD 98647

NUSC Technical Document 6401
10 March 1981

LEVEL II

2

MACSYMA: A Program For Computer Algebraic Manipulation (Demonstrations and Analysis)

R. Drew Drinkard, Jr.
Nancy K. Sulinski
Information Services Department

Naval Underwater Systems Center
Newport, Rhode Island / New London, Connecticut

405918

81 5 04 158

Preface

This document was prepared under NUSC Project No. 771Y00.

The technical reviewer for this document was Mr. Marvin J. Goldstein, Head, Algorithmic Design Section (Code 7122).

Acknowledgments

Appreciation is due to Mr. K. Meals and Dr. E. Cuthill, both of NSRDC, who first introduced us to MACSYMA and who also hosted the course in MACSYMA. Also, we wish to thank Mr. Marvin J. Goldstein who first inspired and encouraged the writing of this document and took the time not only to carefully review the technical aspects of this paper but was instrumental in changing, rearranging, and consolidating the document. Our thanks go also to Dr. Roy Streit, of the Algorithmic Design Section, for his valuable discussions in the preparation of this document. Finally, we must certainly thank Ms. V. Ellen Golden, of Mathlab at MIT, for answering our many inquiries regarding the use of TECO (the editor) and for also taking action to protect our files while we were engaged in preparing these demonstrations.

Reviewed and Approved: 10 March 1981

T G Galib

T. Galib
Information Services Department

The authors of this document are located at the
New London Laboratory, Naval Underwater Systems,
Center, New London, Connecticut 06320.

Table of Contents

MACSYMA: A Program for Computer Algebraic Manipulation (Demonstrations and Analysis)

Introduction

Project MAC's SYmbolic MAnipulation (MACSYMA*) system is a large scale computer program written in the LISP Language with the capability of performing symbolic and numerical mathematical manipulation. The program was developed by the Mathlab Group of the Laboratory for Computer Science at the Massachusetts Institute of Technology and is available to NUSC employees through the NAval Laboratory COmputer Network (NALCON).

The program, MACSYMA, can manipulate algebraic expressions involving constants, variables, and functions. It has the power to differentiate, integrate, take limits, factor polynomials, simplify and make substitutions within symbolic expressions, expand analytic functions (both single and multivariate) in Taylor series, compute Poisson series, and take LaPlace transforms and/or their inverses. It can also plot curves (two-dimensional, three-dimensional, parametric, and contour plots), solve algebraic and ordinary differential equations, and manipulate lists (vector fields, etc.), subscripted variables, tensors, and matrices. MACSYMA can also solve systems of simultaneous linear and nonlinear algebraic and differential equations and allows the user to write specialized programs for manipulating expressions.

To illustrate the utility of MACSYMA as a labor-saving tool in formula simplification, consider reducing a long mathematical expression to a smaller one that requires fewer arithmetic operations to evaluate. Work of this type customarily precedes computer program implementation of mathematical formulas. Doing this may require identifying mathematical subexpressions that are identically zero, simplifying trigonometric expressions using trigonometric identities, or factoring long algebraic expressions. Many of our readers are probably aware from personal experience that this can be an arduous task. However, MACSYMA, with some user assistance, can identify subexpressions that vanish as well as perform other tasks in the simplification process. (See examples C(5), C(23), C(95) and C(96) in Demonstration 2.1 and the Appendix.)

Furthermore, MACSYMA is more than just a symbol manipulator; it has the capability of evaluating expressions. Thus, in addition to representing a particular function symbolically, MACSYMA can be used to evaluate it at a particular point. This ability to combine numerical computation and symbol manipulation into one package is particularly useful since symbolic manipulators are not available in common computer languages, such as FORTRAN, PL/1, ALGOL, and PASCAL.

The following demonstrations are intended to highlight the basic features of the MACSYMA program. Therefore, this is by no means a complete document on the program's capabilities. This technical document is intended to introduce the reader to a powerful symbolic aid for finding computer solutions to mathematical problems in engineering and scientific applications. We also point out some common pitfalls in the use of MACSYMA. For further information on MACSYMA facilities, the user should consult the MACSYMA Reference Manual [1] available from the NUSC Technical Library or the authors.

*Pronounced *maxima*.

1. MACSYMA Syntax

All symbolic expressions are entered via terminal (TTY) as input to MACSYMA much as they would be expressed in FORTRAN, PL/1, ALGOL, or PASCAL, using particular MACSYMA function names when required. These expressions are terminated by the user with either a semicolon (;) or a dollar sign ($) -- not a carriage return. The semicolon indicates to MACSYMA to print out the results of the input expression or respond in some way to a particular request; the dollar sign tells MACSYMA to omit the printing of the result or response to a command. This does not have any bearing, however, on the actual computation done by MACSYMA. Whenever a printed response is desired by the user, MACSYMA prints out the result in a two-dimensional textbook format. Input lines are labeled by MACSYMA consecutively, starting with (C1), (C2), (C3), and so on. The results are labeled (D1), (D2), (D3), and so on. Thus, (D1) is the result of the computations performed by MACSYMA on the user's input string (C1). The D and/or C line numbers may be used to refer to expressions when entering subsequent expressions to MACSYMA. Also, the previous expression printed by MACSYMA can be referenced by a percent (%) sign.

The mathematical operators MACSYMA uses are

\ or !	for factorial,
^ or **	for exponentiation,
*	for multiplication,
.	for non-commutative multiplication,
/	for division,
+	for addition, and
-	for subtraction.

As in FORTRAN, PL/1, ALGOL, and PASCAL, MACSYMA uses an operator precedence scheme similar to that of the languages mentioned to determine the correct order of application of mathematical operators in expressions entered by the user. Parentheses may be used to change the normal order of application of operators.

MACSYMA expressions consist of variables, constants, and operators. Variables (and names of functions and arrays) in MACSYMA consist of any string of numbers and letters, including the character %. Other characters may be included by preceding them with a backslash (\). Variable names are not limited in length, but they must not begin with a number.

Constants are either numbers or one of the special constants, π, e (the base of natural logarithms), or i (the square root of minus one), which are written as %PI, %E, and %I, respectively, to distinguish them from the variables PI, E, and I.

An operator may appear before its arguments (prefix), between its arguments (infix), or after its arguments (postfix). Operators that appear before their arguments are usually functions (such as SIN and COS) or MACSYMA commands (such as INTEGRATE or RATSIMP). The arguments of these prefix operators are usually enclosed in parentheses.

The Introductory Demonstration 1.1 will serve to illustrate the way in which expressions are typed into MACSYMA and the points of MACSYMA syntax discussed so far. The MACSYMA consortium of programs that resides on a large PDP-10 computer also contains a very powerful and versatile operating system command language called Incompatible Time Sharing (ITS). It serves as the parent node for both MACSYMA and TECO. TECO is a large and powerful text editor that can be used both interactively with MACSYMA and independently to create and edit batch files. All the demonstration files contained in this document were created using TECO. The manner in which these files are entered into MACSYMA is described in the introductory demonstration.

```
(C1) BATCH(DRI2,1,DSK,DRINK);

(C2) /*

                              DEMONSTRATION 1.1 - INTRODUCTION

First, we'll type in some expressions in FORTRAN-like syntax.  MACSYMA will reformat the expression and display it in two-
dimensional notation similar to that found in a textbook.  It also assigns the expression a label which may be used in subsequent
commands.  Commands to MACSYMA are written in functional notation.

Exponentiation has higher precedence than Multiplication. */

X*Y^2*Z;
                                                     2
(D2)                                             X Y  Z

(C3) /* Multiplication has higher precedence than Addition.  MACSYMA reorders variables to set the precedence.  */

X+Y*Z;
(D3)                                             Y Z + X

(C4) /* We can use parentheses to change the order of application of operators. */

(X+Y)*Z;
(D4)                                            (Y + X) Z

(C5) /* Division has higher precedence than Subtraction or Addition. */

1/X^2+3;
                                                 1
(D5)                                             -- + 3
                                                  2
                                                 X

(C6) /* Again, we can use parentheses to change the order of application of operators. */

1/(X^2+3);
                                                   1
(D6)                                             ------
                                                  2
                                                 X  + 3

(C7) /* Example of the use of MACSYMA functions, arguments, and constants. */

SIN(3*%PI/2);
(D7)                                              - 1

(C8) COS(2*%PI);
(D8)                                               1

(C9) /* Example of a more complicated expression. */

1/(Y^2+2*X+Z^3)^3-(-4*X+2*Y^2-3*Z^3)/(Y^2+2*X+Z^3)^4+((3*X+2*Y^2-3*Z^3)^2-4*Y)/(Y^2+2*X+Z^3)^2;
                                                    3      2                  3      2       2
                                 1               - 3 Z  + 2 Y  - 4 X    (- 3 Z  + 2 Y  + 3 X)  - 4 Y
(D9)                      ---------------- - -------------------- + ----------------------------
                            3    2      3      3    2       4            3    2       2
                          (Z  + Y  + 2 X)    (Z  + Y  + 2 X)           (Z  + Y  + 2 X)

(C10) /*
```

```
From time to time throughout this demonstration, the reader will notice the setting of certain nouns such as:
                                          DYNAMALLOC[FALSE]:TRUE;
The value inside the square brackets is the default value for that particular noun.  In this particular case, setting
DYNAMALLOC:TRUE will have the effect of increasing the user's storage space automatically as needed without having any questions
asked.  This option should only be used on large batch files because working interactively, the user can allocate up to
five levels of list storage space if needed.
We are now going to show a very useful function command in MACSYMA called DESCRIBE(COMMAND) which is used to describe the
use of most MACSYMA function commands.  We use it here to describe the BATCH and DEMO commands. */

DESCRIBE(BATCH);

DESCRI FASL DSK MACSYM being loaded
Loading done

BATCH(file-specification) reads in and evaluates MACSYMA command
lines from a file - A facility for executing command lines stored on a
disk file rather than in the usual on-line mode. This facility has
several uses, namely to provide a reservoir for working command lines,
for giving error-free demonstrations, or helping in organizing one's
thinking in complex problem-solving situations where modifications may
be done via the PDP-10 TECO file editor.

A batch file consists of a set of MACSYMA command lines, each with its
terminating ; or $, which may be further separated by spaces,
carriage- returns, form-feeds, and the like.

The BATCH function calls for reading in the command lines from the
file one at a time, echoing them on the user console, and executing
them in turn.  Control is returned to the user console only when
serious errors occur or when the end of the file is met.  Of course,
the user may quit out of the file- processing by typing control-G at
any point.

BATCH files may be created using TECO or by use of the STRINGOUT
command.  Do DESCRIBE(STRINGOUT) for details.

NUMER;BATCH FASL has some functions in it that may be useful to users
who use BATCH to run MACSYMA while not being logged in.  Such things
as checking the status of a disowned running macsyma without stopping
it.  Do PRINTFILE(BATCH,INFO,NUMER); for details.

(D10)                                                             DONE

(C11) DESCRIBE(DEMO);

DEMO(file-specification) same as BATCH but pauses after each command
line and continues when a space is typed.

(D11)                                                             DONE

(D12)                                                          BATCH DONE

(C13) ^Z
[DDT]
*
```

2. Manipulating Algebraic Expressions

Instructions for manipulating algebraic expressions are embedded in the demonstration examples. Note that in the demonstration examples, some of the computed expressions returned by MACSYMA are preceded by a /R/ or /T/ following the line label. These serve merely to identify specific types of internal representation and will not be discussed further in this document. Users who are interested in the internal data structures used by MACSYMA in representing, evaluating, and simplifying expressions should see reference [1].

2.1 Simplification of Algebraic Expressions

Algebraic simplification is perhaps the most common process in algebraic manipulation. It is also controversial because it must accommodate two basic points of view: 1) that of a user who seeks to manipulate large and unwieldy expressions, and 2) that of a designer who seeks a useful and efficient system that can be implemented without excessive difficulty. The one common ground between both user and designer is that simplification should change only the form or representation of an expression, it should not alter its value.

An algebraic expression may be transformed into many equivalent forms. Often one of these forms will prove to be more useful than another, but it is not a trivial problem for a designer to implement a symbolic manipulation system that readily converts a given expression into the simplest form, which is useful to the user. It seems clear that a user would prefer a system for algebraic simplification that takes into account contextual information in computing a simplified expression. After all, the simplest form of an expression is dependent upon the user's goals or, in other words, upon the user's context. Thus, algebraic simplification might be described best as a process that transforms algebraic expressions into a form that is a compromise between the designer's and user's points of view.

A very common complaint of new users of MACSYMA (and other algebraic manipulation systems) is that often the expressions obtained as the result of a computation are large and unwieldy to the point of being essentially useless. This becomes less of a problem as the user gains proficiency through experience in using MACSYMA. This point is made in order to prevent the new user from becoming prematurely discouraged from using MACSYMA. For a thorough discussion of the issues surrounding algebraic simplification from the points of view of both the user and designer, see reference [2].

In the following demonstration, we will illustrate simplification and general manipulation of algebraic expressions. This demonstration is subdivided into several separate examples to distinguish between the different expression manipulations that play major roles in the use of MACSYMA.

While looking through demonstration 2.1, it is important to note how the commands in this section can be used to reduce complex algebraic expressions into simpler forms. Also, as will be seen in the example (C(6) and C(7)), "simplifying" procedures sometimes lead to unwanted larger and more cumbersome results than the original expressions. However, as mentioned earlier, proficiency in the use of the system gained through experience will soon show the serious user of MACSYMA that these problems can, to a great extent, be avoided.

Substitution in algebraic expressions is a valuable aid in the comprehension of expressions; however, it provides another example of a user problem in MACSYMA. There are two basic substitution commands in MACSYMA. The appropriate command to use depends on the type of substitution desired. Improper use of the more efficient SUBST command, as opposed to the RATSUBST command, will result in no substitution taking place in the expression at all. This is explained in more detail in the subsection SUBSTITUTION EXAMPLES (in the demonstration).

Another problem that may occur in the use of MACSYMA is often referred to as "intermediate expression swell." This means that expressions in the middle of a calculation become quite large even though the final result may be relatively small.

It is imperative that the reader examines Demonstration 2.1 on "Simplification of Algebraic Expressions" in its entirety. All of the examples in this particular demonstration are very basic and essential to an understanding of what is taking place in all the demonstrations that follow. Most of the other demonstrations in this technical document cover specific applications of MACSYMA and the reader may pursue only demonstrations that are of particular interest without loss of comprehension. Also, since most of the demonstrations contain a number of examples ranging from simple to fairly sophisticated, the reader should look over as many as possible.

```
(C1) BATCH(DRI3,1,DSK,DRINK);

(C2) /*                    DEMONSTRATION 2.1 - SIMPLIFICATION OF ALGEBRAIC EXPRESSIONS
```

Explanation: The Ci lines are commands or expressions typed by the user, while the Di lines are the computer responses. A ';' terminates a command and produces a D-line response, while a '$' terminates a command without producing a D-line response. A '%' is used to refer to the last expression listed. The '%TH(I)' reference operator is used to refer to the Ith previous computation. That is, if the next expression to be computed is D[J], then the expression being referred to is D[J-I]. The ''' character indicates that a function or command is to be displayed but not evaluated or carried out. Also, in many cases a command such as EXPR,EXPAND(i.e.,ARGUMENT,FUNCTION NAME) is equivalent to EXPAND(EXPR). RATSIMP(EXPR) (RATional SIMPlification) is a command which tries to put expressions in a ratio-of-polynomials form and also does simplification.

SIMPLIFICATION EXAMPLES:

```
*/

A+(X*B+B*(A/B-X));
                                                A
(D2)                                  B X + B (- - X) + A
                                                B

(C3) RATSIMP(%);
(D3)                                         2 A

(C4) /* Here, the simplifier has carried out the multiplication and noticed that -BX+BX is equivalent to 0 and A+A is 2A.

We can also use labels to refer to our expressions. */

EXP:(SQRT(R^2+A^2)+A)*(SQRT(R^2+B^2)+B)/R^2-(SQRT(R^2+B^2)+SQRT(R^2+A^2)+B+A)/(SQRT(R^2+B^2)+SQRT(R^2+A^2)-B-A);
                        2    2               2    2            2    2             2    2
              (SQRT(R  + A ) + A) (SQRT(R  + B ) + B)    SQRT(R  + B ) + SQRT(R  + A ) + B + A
(D4)          --------------------------------------- - -------------------------------------
                                 2                          2    2             2    2
                                R                     SQRT(R  + B ) + SQRT(R  + A ) - B - A

(C5) RATSIMP(EXP);
(D5)                                          0

(C6) EXP:1/(Y+X)^3+((X+2)^8-2*Y)/(Y+X)^8;
                                                     8
                                        1     (X + 2)  - 2 Y
(D6)                                 -------- + --------------
                                            3            8
                                     (Y + X)      (Y + X)

(C7) RATSIMP(EXP);
      5         4        2  3        3  2         4              8        7         6         5          4          3          2
      Y  + 5 X Y  + 10 X  Y  + 10 X  Y  + (5 X  - 2) Y + X  + 16 X  + 112 X  + 449 X  + 1120 X  + 1792 X  + 1792 X  + 1024 X + 256
(D7) ----------------------------------------------------------------------------------------------------------------------------
                         8        7         2  6         3  5         4  4         5  3         6  2        7      8
                        Y  + 8 X Y  + 28 X  Y  + 56 X  Y  + 70 X  Y  + 56 X  Y  + 28 X  Y  + 8 X  Y + X

(C8) /* Thus we see that RATSIMP rationally 'simplifies' an expression and all of its subexpressions and returns a quotient of
two polynomials.
```

MACSYMA also has a simplifier for handling expressions involving logs, exponentials, and radicals. This simplifying function is called RADCAN(EXPR) which is an abbreviation for RADical CANonical. Its only major limitation is that it is generally more time consuming with some expressions by comparison with RATSIMP.

Here is a demonstration of the use of RADCAN */

```
(LOG(X^2+X)-LOG(X))^B/LOG(X+1)^(B/2);
                                                    2                 B
                                              (LOG(X  + X) - LOG(X))
(D8)                                          ----------------------
                                                             B/2
                                                 LOG(X + 1)

(C9) RADCAN(%);
                                                             B/2
(D9)                                             LOG(X + 1)

(C10) (%E^((Y+X)/2)+%E^(Y+X))/%E^((Y+X)/2);
                                              Y + X    Y + X
                                            - -----    -----
                                                2        2        Y + X
(D10)                                     %E        (%E      + %E      )

(C11) RADCAN(%);
                                                   Y + X
                                                   -----
                                                     2
(D11)                                            %E       + 1

(C12) (X^(1/5)+X)/X^(1/6);
                                                         1/5
                                                   X + X
(D12)                                              --------
                                                      1/6
                                                     X

(C13) RADCAN(%);
                                                  5/6    1/30
(D13)                                            X    + X

(C14) (X^2+2*X*Y+Y^2)^(1/2)/(X+Y);
                                                      2             2
                                              SQRT(Y  + 2 X Y + X )
(D14)                                         ---------------------
                                                       Y + X

(C15) RADCAN(%);
(D15)                                                    1

(C16) LOG(X^2+6*X+9)/LOG(X+3);
                                                     2
                                                LOG(X  + 6 X + 9)
(D16)                                           -----------------
                                                   LOG(X + 3)

(C17) RADCAN(%);
(D17)                                                    2

(C18) /*
```

EXPANSION EXAMPLES:

The EXPAND(EXPR) command carries out distribution of operators such as multiplication with respect to addition. Many options are available to control the extent of expansion.

Here are two simple examples: */

(A+B)^4;

(D18) $$(B + A)^4$$

(C19) EXPAND(%);

(D19) $$B^4 + 4 A B^3 + 6 A^2 B^2 + 4 A^3 B + A^4$$

(C20) (X+Y)^2*(A+B)^2;

(D20) $$(B + A)^2 (Y + X)^2$$

(C21) EXPAND(%);

(D21) $$B^2 Y^2 + 2 A B Y^2 + A^2 Y^2 + 2 B^2 X Y + 4 A B X Y + 2 A^2 X Y + B^2 X^2 + 2 A B X^2 + A^2 X^2$$

(C22) /* On the other hand, the factored form of expressions being smaller than the expanded form, you may wish to factor an expression using the FACTOR(EXPR) command in MACSYMA.

Now consider rationally simplifying and factoring an expanded expression. */

%TH(3)/%TH(1);

(D22) $$\frac{B^4 + 4 A B^3 + 6 A^2 B^2 + 4 A^3 B + A^4}{B^2 Y^2 + 2 A B Y^2 + A^2 Y^2 + 2 B^2 X Y + 4 A B X Y + 2 A^2 X Y + B^2 X^2 + 2 A B X^2 + A^2 X^2}$$

(C23) FACTOR(RATSIMP(%));

(D23) $$\frac{(B + A)^2}{(Y + X)^2}$$

(C24) /* RATSIMP has cancelled out the factor (A+B)^2. Also, note the nesting of commands in the above MACSYMA statement.

The next example will demonstrate the use of several other MACSYMA commands which are used in manipulating and simplifying algebraic expressions.

The command MULTTHRU(EXPR) is used to multiply each summand of a factor of an expression by the other factors of the expression. That is, the expression consists of f1*f2*...*fn where at least one factor, say f1, is a sum of terms. Each term in that sum is multiplied by the other factors in the product. MULTTHRU does not expand exponentiated sums. This function is the fastest way to distribute products(commutative or noncommutative) over sums.

The MACSYMA command COMBINE(EXPR) simplifies the sum represented in the expression by combining terms with the same denominator into a single term.

This example demonstrates the combined use of these MACSYMA commands in conjunction with EXPAND, RATSIMP, and FACTOR. */

(X+Y+Z)^2/Y;

(D24) $$\frac{(Z + Y + X)^2}{Y}$$

(C25) RATSIMP(%);

(D25) $$\frac{Z^2 + (2 Y + 2 X) Z + Y^2 + 2 X Y + X^2}{Y}$$

(C26) MULTTHRU(%);

(D26) $$\frac{Z^2}{Y} + \frac{(2 Y + 2 X) Z}{Y} + Y + \frac{X^2}{Y} + 2 X$$

(C27) EXPAND(%);

(D27) $$\frac{Z^2}{Y} + \frac{2 X Z}{Y} + 2 Z + Y + \frac{X^2}{Y} + 2 X$$

(C28) COMBINE(%);

COMBF FASL DSK MAXOUT being loaded
Loading done

(D28) $$\frac{Z^2 + 2 X Z + X^2}{Y} + 2 Z + Y + 2 X$$

(C29) FACTOR(%);

(D29) $$\frac{(Z + Y + X)^2}{Y}$$

(C30) /* We now have the expression that we started with, as intended, to demonstrate the various commands discussed up to this point.

EXTRACTION OF PARTS EXAMPLES:

Very useful tools in MACSYMA are the Part functions which make it possible to reference or replace any part of any MACSYMA expression. A part of a displayed expression is referred to by a set of indices which are non-negative integers. For example, in exponentiation, the base is considered part 1 and the exponent part 2. In a sum or product the Ith term or factor is part I. In any expression the main operator is part 0. For -X the 0th part is -, for A^B it is ^, for EXPAND((A+B)^4); it is EXPAND, etc.

To extract the 4th term(4) in the denominator(2) of the first expression in the expansion example, first draw a box around it to see if that is the term we want, and if so extract it. This is accomplished with the commands DPART(EXPR,part specifications) for the box and PART(EXPR,part specifications) for the extraction. */

```
DPART(%TH(8),2,4);

BOX FASL DSK MAXOUT being loaded
Loading done
                              4        3      2  2      3     4
                              B  + 4 A B  + 6 A  B  + 4 A  B + A
(D30)      ------------------------------------------------------------------------
                                 **********
             2  2          2    2  2    *  2    *                    2        2  2          2    2  2
             B  Y  + 2 A B Y  + A  Y  + *2 B  X Y* + 4 A B X Y + 2 A  X Y + B  X  + 2 A B X  + A  X
                                 **********

(C31) PART(%TH(9),2,4);
                                                       2
(D31)                                              2 B  X Y

(C32) /* First we'll 'box' the Y in the following expression: */

A=Z+W^2+X/Y^2;
                                                   X     2
(D32)                                      A = Z + -- + W
                                                    2
                                                   Y

(C33) /* The 1st '2' refers to the right-hand side of an equation expression, the 2nd '2' refers to the 2nd term of this side of
the equation, the 3rd '2' refers to the denominator of this term, and the '1' refers to the base of the exponent which is Y. */

DPART(%,2,2,2,1);
                                                   X      2
(D33)                                     A = Z + ---- + W
                                                  ***2
                                                  *Y*
                                                  ***

(C34) /* Next we'll 'box' the W in the same expression. The 1st '2' again refers to the right-hand side of the equation, the
'3' refers to the 3rd term of this side of the expression, and the '1' refers to the base of the exponent which is W. */

DPART(%TH(2),2,3,1);
                                                        ***2
                                                  X     
(D34)                                     A = Z + -- + *W*
                                                   2    ***
                                                  Y

(C35) /*
```

```
To extract rational coefficients RATCOEF(EXPR,VAR,N), which returns the coefficient of the term VAR^N in the expression
EXPR, is used.  N may be omitted if it is 1.  Here the coefficient of A^4 in (A+B)^8 is found. */

(A+B)^8;
                                                              8
(D35)                                                  (B + A)

(C36) EXPAND(%);
                    8         7        2  6       3  5       4  4       5  3       6  2       7      8
(D36)              B  + 8 A B  + 28 A  B  + 56 A  B  + 70 A  B  + 56 A  B  + 28 A  B  + 8 A  B + A

(C37) RATCOEF((A+B)^8,A^4);
                                                              4
(D37)                                                     70 B

(C38) /* In the next expression, we want to extract the expanded coefficient of X+A. */

(X+A)*(Y+3)^2-(X+A)*COS(A)^2-(Y+C)^3=0;
                                         3                  2       2
(D38)                           - (Y + C)  + (X + A) (Y + 3)  - COS (A) (X + A) = 0

(C39) EXPAND(%);
                  3       2         2       2             2                  2                3            2
(D39)          - Y  + X Y  - 3 C Y  + A Y  + 6 X Y - 3 C  Y + 6 A Y - COS (A) X + 9 X - C  - A COS (A) + 9 A = 0

(C40) RATCOEF(LHS(%),X+A);
                                                      2            2
(D40)                                                Y  + 6 Y - COS (A) + 9

(C41) /* Note the left-hand-side operator(LHS) had to be used since equations are not rational expressions.  The coefficient of
X+A is not immediately evident in the expanded form of the expression.

Extracting real and imaginary parts is shown next.  Recall that %E is the base of the natural log and %I is the SQRT(-1). */

%E^(%I*OMEGA);
                                                         %I OMEGA
(D41)                                                  %E

(C42) /* The noun DEMOIVRE[FALSE] if TRUE will cause an expression of the form %E^[A+B*%I] to become %E^A*(COS(B)+%I*SIN(B)) if B
is free of %I. */

DEMOIVRE:TRUE;
(D42)                                                     TRUE

(C43) %E^(%I*OMEGA);
(D43)                                          %I SIN(OMEGA) + COS(OMEGA)

(C44) REALPART(%TH(3));

RPART FASL DSK MACSYM being loaded
Loading done
(D44)                                                  COS(OMEGA)

(C45) IMAGPART(%TH(4));
(D45)                                                  SIN(OMEGA)

(C46) DEMOIVRE:FALSE;
(D46)                                                    FALSE
```



```
(C47) /*
```

```
Consider another expression: */

(A+%I*B)/(D+%I*E);
                                                    %I B + A
(D47)                                               --------
                                                    %I E + D

(C48) IMAGPART(%);
                                                   B D - A E
(D48)                                              ---------
                                                     2    2
                                                    E  + D

(C49) REALPART(%TH(2));
                                                   B E + A D
(D49)                                              ---------
                                                     2    2
                                                    E  + D

(C50) /* These last two results are obtained by multiplying the numerator and denominator of the expression in D47 by the complex
conjugate of the denominator and then extracting the real and imaginary parts of the resulting expression.
```

SUBSTITUTION EXAMPLES:

Often the user will want to have access to a finer level of control over her/his expressions. MACSYMA has several commands which allow the user to make certain substitutions/transformations in her/his expressions. One important command of this type in MACSYMA is SUBST(A,B,C) which substitutes an expression A for subexpression B in expression C. The resulting expression will be mathematically equivalent to the original expression if A is mathematically equivalent to B.

The main limitation with the use of SUBST is that it can only substitute for expressions that constitute an extractable part of the larger expression they are contained in. In the case where the subexpression that the user wants to substitute for is embedded in the expression, a more powerful tool is needed. The command RATSUBST(A,B,C) is helpful in these cases.

```
We will first demonstrate the use of SUBST. */

Q^2*NO+E*M;
                                           2
(D50)                                  NO Q  + E M

(C51) /* Let's substitute T for Q^2*NO and R for E*M in this expression using SUBST.  In this case both Q^2*NO and E*M are
examples of extractable subexpressions. */

SUBST(T,Q^2*NO,%);
(D51)                                   T + E M

(C52) SUBST(R,E*M,%);
(D52)                                    T + R

(C53) /* However, suppose that we have an expression like */

Q^2*NO*E/M/%I/W;
                                                 2
                                       %I E NO Q
(D53)                                - -----------
                                           M W

(C54) /* Now, using SUBST again, we will attempt to substitute T for E*NO in this expression. */

SUBST(T,E*NO,%);
                                                 2
                                       %I E NO Q
(D54)                                - -----------
                                           M W

(C55) /* The substitution was not made because the function SUBST will not handle a subexpression such as E*NO because it is
embedded as a factor in the numerator.  To effect the substitution the command RATSUBST must be used. */

RATSUBST(T,E*NO,%TH(2));
                                            2
                                       %I Q  T
(D55)                                 - -------
                                          M W

(C56) /*
```

```
Similarly, if we want to substitute R for E/M in the original expression, we will have to use RATSUBST to effect
the substitution because E/M is embedded as a rational factor of the entire expression. */

SUBST(R,E/M,%TH(3));
                                                                   2
                                                         %I E NO Q
(D56)                                                  - ----------
                                                            M W

(C57) RATSUBST(R,E/M,%);
                                                                2
                                                         %I NO Q  R
(D57)                                                  - ----------
                                                             W

(C58) /* In the next expression we will attempt to substitute R for X+Y using SUBST. */

2*(X+Y+Z);
(D58)                                                 2 (Z + Y + X)

(C59) SUBST(R,X+Y,%);
(D59)                                                 2 (Z + Y + X)

(C60) /*  The substitution did not take place using SUBST because X+Y is an embedded subexpression of a factor (X+Y+Z).  However,
when we attempt the substitution using RATSUBST the substitution takes place.  */

RATSUBST(R,X+Y,%);
(D60)                                                   2 Z + 2 R

(C61) /* Next, consider substituting W[PE]^2*EP[0] for Q^2*NO/M in the following expression.  The square brackets "[]" are used to
indicate subscripts in MACSYMA commands. */

Q^2*NO*E/M/%I/W;
                                                                   2
                                                         %I E NO Q
(D61)                                                  - ----------
                                                            M W

(C62) RATSUBST(W[PE]^2*EP[0],Q^2*NO/M,%);
                                                                   2
                                                        %I EP  E W
                                                             0    PE
(D62)                                                 - ------------
                                                             W

(C63) /*
```

```
Another useful command in MACSYMA which allows one to make a substitution in a specific part of an expression without having
to extract, substitute, and replace that part of the expression is called SUBSTPART(x,exp,n1,...,nk) which substitutes x for
the subexpression of the expression exp picked out by the rest of the arguments as in PART.  It returns the new value of exp.
A useful command used in conjunction with any of the Part functions is PIECE.  PIECE holds the part of the last expression
selected when using the Part functions while performing an operation on that piece of the expression specified by the arguments
n1,...,nk.  For example, consider the following expression: */

A+B/(X*(Y+(A+B)*X)+1);
                                                         B
(D63)                                          --------------------- + A
                                               X (Y + (B + A) X) + 1

(C64) SUBSTPART(MULTTHRU(PIECE),%,1,2,1);
                                                         B
(D64)                                           -------------------- + A
                                                                 2
                                                X Y + (B + A) X  + 1

(C65) /* The thing being substituted in the first expression above is a distribution of products over sums in the first term of
the denominator of the first term of the expression.
Consider the following expression: */

(A*X+Y)^2+(A*U+V)^2;
                                                        2            2
(D65)                                           (Y + A X)  + (V + A U)

(C66) /* Expand this expression. */

EXPAND(%);
                                         2              2  2    2               2  2
(D66)                                   Y  + 2 A X Y + A  X  + V  + 2 A U V + A  U

(C67) /* Now, try factoring this expanded result to get back what we started with. */

FACTOR(%);
                                         2              2  2    2               2  2
(D67)                                   Y  + 2 A X Y + A  X  + V  + 2 A U V + A  U

(C68) /* No success!  This is one of the many peculiarities in MACSYMA which is not a real limitation.  Experience will soon
teach the serious user of MACSYMA that there are ways to work around these problems.  We will provide a demonstration for this
case.  Let's first factor just the part 'boxed' in the expression below: */

DPART(%TH(2),[1,2,3]);
                                       ****             *******
                                       * 2*  *********  * 2  2*    2               2  2
(D68)                                  *Y  * + *2 A X Y* + *A  X  * + V  + 2 A U V + A  U
                                       ****  *********  *******

(C69) /* Then, let's factor the remaining three terms of this expression: */

DPART(%TH(3),[4,5,6]);
                                                          ****            *******
                                     2              2  2  * 2*  *********  * 2  2*
(D69)                               Y  + 2 A X Y + A  X  + *V  * + *2 A U V* + *A  U *
                                                          ****  *********  *******

(C70) /*
```

```
*/

SUBSTPART(FACTOR(PIECE),%TH(4),[1,2,3]);
```

(D70) $(Y + A X)^2 + V^2 + 2 A U V + A^2 U^2$

```
(C71) SUBSTPART(FACTOR(PIECE),%,[2,3,4]);
```

(D71) $(Y + A X)^2 + (V + A U)^2$

```
(C72) /* We have substituted into the expression in D66 only the part currently being factored until the factoring of the entire
expression is complete.


FACTORING EXAMPLE:

*/

X^99+1;
```

(D72) $X^{99} + 1$

```
(C73) FACTOR(%);
```

(D73) $(X + 1) (X^2 - X + 1) (X^6 - X^3 + 1) (X^{10} - X^9 + X^8 - X^7 + X^6 - X^5 + X^4 - X^3 + X^2 - X + 1)$

$(X^{20} + X^{19} - X^{17} - X^{16} + X^{14} + X^{13} - X^{11} - X^{10} - X^9 + X^7 + X^6 - X^4 - X^3 + X + 1)$

$(X^{60} + X^{57} - X^{51} - X^{48} + X^{42} + X^{39} - X^{33} - X^{30} - X^{27} + X^{21} + X^{18} - X^{12} - X^9 + X^3 + 1)$

```
(C74) /*
```

```
PARTIAL FRACTION EXPANSION EXAMPLE:

We could apply RATSIMP to the following expression to put it in a ratio-of-polynomials form which might improve the
overall execution time.  However, we will omit that step to keep the expression closer to standard form.  */

(X+2)/(X+3)/(X+B)/(X+C)^2;
                                                    X + 2
(D74)                                   ----------------------
                                                              2
                                        (X + 3) (X + B) (X + C)

(C75) /* Now, we will take a partial fraction expansion of this expression. */

PARTFRAC(%,X);

PFRAC FASL DSK MAXOUT being loaded
Loading done
                              2
                             C  - 4 C - B + 6                                                     C - 2
(D75) - ---------------------------------------------------------------------- - -----------------------------------
          4               3     2             2         2              2           2                             2
        (C  + (- 2 B - 6) C  + (B  + 12 B + 9) C  + (- 6 B  - 18 B) C + 9 B ) (X + C)   (C  + (- B - 3) C + 3 B) (X + C)

                                        B - 2                                              1
                + ------------------------------------------------ - -----------------------------------------------
                              2                2     3     2                     2
                  ((B - 3) C  + (6 B - 2 B ) C + B  - 3 B ) (X + B)   ((B - 3) C  + (18 - 6 B) C + 9 B - 27) (X + 3)

(C76) /* This does not look like a partial fraction expansion because factors have been multiplied out.  To convert this
into an expression closer to standard form we will define a function in MACSYMA called FACTERM(X), which will factor
subexpression factors, X, that are sums.  Note that the X factors are the coefficients of (X+3),(X+B), and (X+C) in the
denominator of the above expression.  The function MAP(FACTOR,X) will be used to map the function FACTOR onto each X term of this
very large expression rather than to attempt to factor the entire expression all at once which could possibly cause intermediate
expression swell to the point of exhaustion of list storage space during computation.  */

FACTERM(X):=MAP(FACTOR,X);
(D76)                                     FACTERM(X) := MAP(FACTOR, X)

(C77) FACTERM(%TH(2));
              2
             C  - 4 C - B + 6               C - 2                       B - 2                      1
(D77)   - ---------------------- - ----------------------- + ----------------------- - -----------------------
                 2       2                                2                  2                            2
          (C - 3)  (C - B)  (X + C)  (C - 3) (C - B) (X + C)   (B - 3) (C - B)  (X + B)  (B - 3) (C - 3)  (X + 3)

(C78) /* Now, we will re-factor this expression to obtain the original result as verification of the expansion. */

FACTOR(%);
                                                    X + 2
(D78)                                   ----------------------
                                                              2
                                        (X + 3) (X + B) (X + C)

(C79) /*
```

TRIGONOMETRIC SIMPLIFICATION EXAMPLES:

Much of trigonometric simplification in MACSYMA assumes that the user has a reasonable knowledge of the basic trigonometric identities and formulas so that appropriate substitutions can be made in the expressions using the MACSYMA commands SUBST and RATSUBST. However, there are some commands in MACSYMA which will greatly aid the user in simplifying trigonometric expressions. The function command TRIGEXPAND expands trigonometric and hyperbolic functions of sums of angles and of multiples of angles occurring in an expression. To enhance user control of simplification, this function expands only one level at a time, expanding sums of angles or multiples of angles. To obtain a full expansion into sines and cosines immediately, the switch TRIGEXPAND[FALSE]:TRUE must be set. The setting of the switch HALFANGLES[FALSE]:TRUE causes half-angles to be simplified away. Here are a few simple examples: */

```
X+SIN(3*X)/SIN(X);
                                        SIN(3 X)
(D79)                                   -------- + X
                                         SIN(X)

(C80) TRIGEXPAND:TRUE;
(D80)                                     TRUE

(C81) X+SIN(3*X)/SIN(X);

TRIGEX FASL DSK MAXOUT being loaded
Loading done
                                     2              3
                                3 COS (X) SIN(X) - SIN (X)
(D81)                           -------------------------- + X
                                          SIN(X)

(C82) FACTOR(%);
                                      2            2
(D82)                           - (SIN (X) - 3 COS (X) - X)

(C83) TRIGEXPAND:FALSE;
(D83)                                    FALSE

(C84) SIN(10*X+Y);
(D84)                                SIN(Y + 10 X)

(C85) TRIGEXPAND(%);
(D85)                     COS(10 X) SIN(Y) + SIN(10 X) COS(Y)

(C86) COS(X+Y);
(D86)                                 COS(Y + X)

(C87) TRIGEXPAND:TRUE;
(D87)                                    TRUE

(C88) COS(X+Y);
(D88)                          COS(X) COS(Y) - SIN(X) SIN(Y)

(C89) TRIGEXPAND:FALSE;
(D89)                                    FALSE

(C90) /*
```

```
*/

SIN(X/2)/SEC(X/2);
                                                 X
                                             SIN(-)
                                                 2
(D90)                                        ------
                                                 X
                                             SEC(-)
                                                 2

(C91) HALFANGLES:TRUE;
(D91)                                         TRUE

(C92) SIN(X/2)/SEC(X/2);

LOGARC FASL DSK MACSYM being loaded
Loading done
                                        SQRT(1 - COS(X))
(D92)                                   ----------------
                                                    X
                                         SQRT(2) SEC(-)
                                                    2

(C93) RATSUBST(1/COS(X/2),SEC(X/2),%);
                                 SQRT(1 - COS(X)) SQRT(COS(X) + 1)
(D93)                            ---------------------------------
                                                 2

(C94) HALFANGLES:FALSE;
(D94)                                        FALSE

(C95) /* The function TRIGREDUCE(EXP,VAR) in MACSYMA combines products and powers of trigonometric and hyperbolic SINs and
COSs of VAR, in expression EXP, into multiple angle expressions of VAR.  It also tries to eliminate these functions when they
occur in denominators.  If VAR is omitted, then all variables in EXP are used. */

%TH(13);
                                          2            2
(D95)                               - (SIN (X) - 3 COS (X) - X)

(C96) TRIGREDUCE(%);

TRGRED FASL DSK MACSYM being loaded
Loading done

SCHATC FASL DSK MACSYM being loaded
Loading done

BINOML FASL DSK MAXOUT being loaded
Loading done
(D96)                                 2 COS(2 X) + X + 1

(C97) /*
```

```
The trigonometric simplification routines will use declared information in some simple cases.  For example: */

COS(N*%PI/2);
                                                   %PI N
(D97)                                          COS(-----)
                                                     2

(C98) DECLARE(N,ODD);
(D98)                                            DONE

(C99) COS(N*%PI/2);
(D99)                                             0

(C100) DECLARE(M,EVEN);
(D100)                                           DONE

(C101) COS(M*%PI/2);
                                                    M/2
(D101)                                         (- 1)

(C102) /* There are a number of ways the user may invoke identities such as: */

SIN(X)^2=1-COS(X)^2;
                                            2             2
(D102)                                   SIN (X) = 1 - COS (X)

(C103) %TH(8);
                                            2             2
(D103)                               - (SIN (X) - 3 COS (X) - X)

(C104) /* This is one way of invoking identity D102 on the above expression: */

%,SIN(X)^2=1-COS(X)^2;
                                                  2
(D104)                                       4 COS (X) + X - 1

(C105) /* To transform */

SIN(X)^4;
                                                 4
(D105)                                        SIN (X)

(C106) /* using D102, one needs the added power of RATSUBST since SIN(X)^2 is embedded in D105. */

RATSUBST(1-COS(X)^2,SIN(X)^2,%);
                                           4            2
(D106)                                  COS (X) - 2 COS (X) + 1

(C107) /* Take a look at this next expression: */

SIN(X)^4+(SIN(X)^2+COS(X)^2)/COS(X)^2;
                                                 2          2
                                           4  SIN (X) + COS (X)
(D107)                                  SIN (X) + -----------------
                                                        2
                                                     COS (X)

(C108) /*
```

```
First we will try TRIGREDUCE on it! */

TRIGREDUCE(%);
                                   COS(4 X) - 4 COS(2 X) + 3      2
(D108)                             -------------------------- + SEC (X)
                                               8

(C109) /* This did not result in the expression being reduced to its simplest form.  Now we will try TRIGREDUCE on
just the second term of the expression and substitute the result back into the expression. */

SUBSTPART(TRIGREDUCE(PIECE),%TH(2),2);
                                               4        2
(D109)                                      SIN (X) + SEC (X)

(C110) /* That result was better!  Now, let's try the following identity: */

1-SIN(X)^2=COS(X)^2;
                                                  2         2
(D110)                                     1 - SIN (X) = COS (X)

(C111) /* on an expression where: */

COS(X)^2;
                                                      2
(D111)                                             COS (X)

(C112) /* is so embedded that it is not easily extractable from the expression.  This is the expression! */

COS(X)^3*SIN(X)+COS(X)*SIN(X)^3;
                                                3         3
(D112)                              COS(X) SIN (X) + COS (X) SIN(X)

(C113) RATSUBST(1-SIN(X)^2,COS(X)^2,%);
(D113)                                        COS(X) SIN(X)

(C114) /* It should be pointed out that while TRIGREDUCE works fine in most cases, it is, nevertheless, costly to use in terms
of time and storage space because it requires loading of at least three large MACSYMA library routines into core.  */

(D114)                                             BATCH DONE

(C115) ^Z
[DDT]
*
```

2.2 Calculus Operations

For the reader who is interested in symbolic calculus operations, the basic sections to read are DERIVATIVE EXAMPLES, INDEFINITE INTEGRAL EXAMPLES, and DEFINITE INTEGRAL EXAMPLES. The sections on vector differential calculus and integration by special contours and residue theorem deal with special areas of application that the reader can omit without any loss of continuity.

Although MACSYMA is a useful tool designed to aid mathematicians, engineers, and scientists, it is not a "magical device" that can solve symbolically all mathematical problems presented to it. For example, some elliptic integrals must be evaluated numerically; this will, of course, require a little more interaction between the user and MACSYMA. Furthermore, although MACSYMA has a special package of routines in the SHARE directory for handling symbolic definite integration of special functions, MACSYMA's present capability in this area is limited. However, work is currently being done at MIT to improve this capability.

In general, handling definite integrals symbolically presents execution time and main memory storage problems. This is mainly due to the rather large number of MACSYMA library routines required to evaluate them. A point worth mentioning here is that while some or all of the user's space can be freed by the user, it is not possible to free space that has been taken up in core by MACSYMA library routines.

(C1) BATCH(DRI4,1,DSK,DRINK);

(C2) /*

DEMONSTRATION 2.2 - CALCULUS OPERATIONS

DERIVATIVE EXAMPLES:

Below are examples of the DIFF(EXP,VAR,N) operator in MACSYMA which performs symbolic differentiation. It calculates the Nth derivative of expression EXP with respect to variable VAR. Here is an expression to illustrate: */

X^2*SIN(X)+Y*X+%E^X^2/X+COSH(X);

HYPER FASL DSK MAXOUT being loaded
Loading done

(D2) $$X\ Y + X^{2}\ \mathrm{SIN}(X) + \mathrm{COSH}(X) + \frac{\%E^{X^{2}}}{X}$$

(C3) /* We will calculate the 1st derivative of this expression with respect to X. */

DIFF(%,X);

(D3) $$Y + \mathrm{SINH}(X) + 2\ X\ \mathrm{SIN}(X) + X^{2}\ \mathrm{COS}(X) - \frac{\%E^{X^{2}}}{X^{2}} + 2\ \%E^{X^{2}}$$

(C4) /* The 2nd derivative of this same expression with respect to X is */

DIFF(%TH(2),X,2);

(D4) $$- X^{2}\ \mathrm{SIN}(X) + 2\ \mathrm{SIN}(X) + \mathrm{COSH}(X) + 4\ X\ \mathrm{COS}(X) + 4\ X\ \%E^{X^{2}} - \frac{2\ \%E^{X^{2}}}{X} + \frac{2\ \%E^{X^{2}}}{X^{3}}$$

(C5) /* The 5th derivative of this same expression with respect to X is */

DIFF(%TH(3),X,5);

(D5) $$\mathrm{SINH}(X) + 10\ X\ \mathrm{SIN}(X) + X^{2}\ \mathrm{COS}(X) - 20\ \mathrm{COS}(X) + 32\ X^{4}\ \%E^{X^{2}} + 80\ X^{2}\ \%E^{X^{2}} - \frac{60\ \%E^{X^{2}}}{X^{2}} + \frac{120\ \%E^{X^{2}}}{X^{4}} - \frac{120\ \%E^{X^{2}}}{X^{6}} + 40\ \%E^{X^{2}}$$

(C6) /* The derivative expression can be displayed before evaluating it by prefixing the DIFF operator with a "'" such as: */

'DIFF(SIN(X)+X^3+2*X^2,X);

(D6) $$\frac{d}{dX}\ (\mathrm{SIN}(X) + X^{3} + 2\ X^{2})$$

(C7) /*

```
Then, we can evaluate this derivative expression: */

EV(%,DIFF);
                                              2
(D7)                            COS(X) + 3 X  + 4 X

(C8) /* The more complete operator specification DIFF(EXP,V1,N1,V2,N2,...) differentiates expression EXP with respect to each
variable Vi to the Nith derivative. */

'DIFF(Y^2+X*Y+X^2,X,2,Y,1);
                                   3
                                  d        2            2
(D8)                            ------ (Y  + X Y + X )
                                  2
                                dX  dY

(C9) /* Now, let's evaluate this derivative expression: */

EV(%,DIFF);
(D9)                                      0

(C10) /* Next, we will define a function Z of two variables X and Y: */

Z(X,Y):=%E^SIN(X)*TAN(X*Y);
                                      SIN(X)
(D10)                        Z(X, Y) := %E       TAN(X Y)

(C11) /* We will now display and evaluate the 1st partial derivative of this function Z(X,Y) with respect to X: */

'DIFF(Z(X,Y),X);
                                d     SIN(X)
(D11)                           -- (%E        TAN(X Y))
                                dX

(C12) EV(%,DIFF);
                          SIN(X)                  SIN(X)      2
(D12)               COS(X) %E       TAN(X Y) + %E       Y SEC (X Y)

(C13) /* We can also evaluate this partial derivative at a point such as (X,Y)=(%PI/4,1): */

EV(%,X=%PI/4,Y=1);
                                      SQRT(2)
                                      -------        SQRT(2)
                                         2           -------
                                SQRT(2) %E              2
(D13)                           ----------------- + 2 %E
                                        2

(C14) /* Next, we will display and evaluate the first partial derivative of the function Z(X,Y) with respect to Y: */

'DIFF(Z(X,Y),Y);
                                d     SIN(X)
(D14)                           -- (%E        TAN(X Y))
                                dY

(C15) EV(%,DIFF);
                                    SIN(X)    2
(D15)                           X %E       SEC (X Y)

(C16) /*
```

```
Let's also evaluate this partial derivative at the same point (X,Y)=(%PI/4,1): */

EV(%,X=%PI/4,Y=1);
                                                     SQRT(2)
                                                     -------
                                                        2
                                                   %E         %PI
(D16)                                              -------------
                                                         2

(C17) /* Let's now define another function Z(X,Y): */

Z(X,Y):=LOG(SIN(X^2*Y^2-1));
                                                     2  2
(D17)                              Z(X, Y) := LOG(SIN(X  Y  - 1))

(C18) /* The form of the operator DIFF(EXP) with no other arguments gives the 'total differential'. */

DIFF(Z(X,Y));

DIFF2 FASL DSK MAXOUT being loaded
Loading done
                      2         2  2                        2      2  2
                   2 X  Y COS(X  Y  - 1) DEL(Y)     2 X Y  COS(X  Y  - 1) DEL(X)
(D18)              ---------------------------- + ----------------------------
                              2  2                           2  2
                         SIN(X  Y  - 1)                 SIN(X  Y  - 1)

(C19) /* We can simplify this expresion a bit by using substitution: */

RATSUBST(COT(X^2*Y^2-1),COS(X^2*Y^2-1)/SIN(X^2*Y^2-1),%);
                      2         2  2                     2      2  2
(D19)              2 X  Y COT(X  Y  - 1) DEL(Y) + 2 X Y  COT(X  Y  - 1) DEL(X)

(C20) /* Next, we define a function W of 3 variables X,Y,Z: */

W(X,Y,Z):=(X*Y)/Z;
                                                        X Y
(D20)                                       W(X, Y, Z) := ---
                                                         Z

(C21) /* Then we calculate the 'total differential' of W(X,Y,Z): */

DIFF(W(X,Y,Z));
                                      X Y DEL(Z)   X DEL(Y)   Y DEL(X)
(D21)                               - ---------- + -------- + --------
                                          2           Z          Z
                                         Z

(C22) /* The period operator "." is used for noncommutative multiplication which can be used for vector or matrix multiplication
or for defining special operators.  In the following example we want the matrices A,B,C,D dependent on variable T: */

DEPENDENCIES(A(T),B(T),C(T),D(T));
(D22)                                    [A(T), B(T), C(T), D(T)]

(C23) /*
```

```
Here is an expression using the "." operator and these 4 matrices: */

A.B+C.D;

MDOT FASL DSK MACSYM being loaded
Loading done
(D23)                                          C . D + A . B

(C24) /* Now, we will calculate the 1st derivative of this expression with respect to T: */

DIFF(%,T);
                                        dC             dD    dA             dB
                                        -- . D + C . -- + -- . B + A . --
(D24)                                   dT             dT    dT             dT

(C25) /* We are now going to free some storage space before proceeding with our demonstration. */

KILL(ALL);                                             DONE
(D0)

(C1) DYNAMALLOC:TRUE;                                  TRUE
(D1)

(C2) /*
```

```
INDEFINITE INTEGRAL EXAMPLES:

The following examples demonstrate the display and evaluation of indefinite integrals of real functions using MACSYMA's
INTEGRATE(EXP,VAR) operator.  This operator symbolically integrates the expression EXP with respect to the variable VAR. */

'INTEGRATE(%E^X/(%E^X+2),X);
                                                          /      X
                                                          [    %E
(D2)                                                      I -------- dX
                                                          ]    X
                                                          / %E   + 2

(C3) EV(%,INTEGRATE);

SIN FASL DSK MACSYM being loaded
Loading done

SININT FASL DSK MACSYM being loaded
Loading done

SCHATC FASL DSK MACSYM being loaded
Loading done
                                                                 X
(D3)                                                      LOG(%E   + 2)

(C4) 'INTEGRATE(1/(X*LOG(X)),X);
                                                         /
                                                         [       1
(D4)                                                     I -------- dX
                                                         ] X LOG(X)
                                                         /

(C5) EV(%,INTEGRATE);
(D5)                                                     LOG(LOG(X))

(C6) 'INTEGRATE(%E^(A*U)*SIN(N*U),U);
                                                       /
                                                       [    A U
(D6)                                                   I %E     SIN(N U) dU
                                                       ]
                                                       /

(C7) EV(%,INTEGRATE);
                                                   A U
                                                 %E    (A SIN(N U) - N COS(N U))
(D7)                                             -------------------------------
                                                               2    2
                                                              N  + A

(C8) ASSUME(N>=0);
(D8)                                                         [N >= 0]

(C9) /*
```

```
$/
'INTEGRATE(U^N*LOG(U),U)$
                                          /
                                          [  N
(D9)                                      I U  LOG(U) dU
                                          ]
                                          /

(C10) EV(%,INTEGRATE)$
                                      N + 1             N + 1
                                     U      LOG(U)     U
(D10)                                -------------- - --------
                                         N + 1               2
                                                      (N + 1)

(C11) ASSUME(N>A,A>0)$
(D11)                                     [N > A, A > 0]

(C12) 'INTEGRATE(%E^(A*U)*COSH(N*U),U)$
                                     /
                                     [    A U
(D12)                                I %E     COSH(N U) dU
                                     ]
                                     /

(C13) EV(%,INTEGRATE)$
RISCH FASL DSK MACSYM being loaded
Loading done
                                      (N + A) U       (A - N) U
                                    %E              %E
                                    ------------ + -----------
                                       N + A          A - N
(D13)                               ---------------------------
                                                 2

(C14) FACTOR(RATSIMP(%))$
                                 2 N U        2 N U            A U - N U
                            (N %E      - A %E      - N - A) %E
(D14)                       ---------------------------------------------
                                         2 (N - A) (N + A)

(C15) ASSUME((4*B^2-4*A^2)>0)$
                                            2      2
(D15)                                    [4 B  > 4 A ]

(C16) 'INTEGRATE(1/(A+B*SIN(U)),U)$
                                      /
                                      [       1
(D16)                                 I ------------ dU
                                      ] B SIN(U) + A
                                      /

(C17) /*
```

```
*/
EV(%,INTEGRATE);
                            2 A SIN(U)              2     2
                            ---------- - 2 SQRT(B  - A ) + 2 B
                            COS(U) + 1
                        LOG(----------------------------------)
                            2 A SIN(U)              2     2
                            ---------- + 2 SQRT(B  - A ) + 2 B
                            COS(U) + 1
(D17)                   --------------------------------------
                                             2    2
                                        SQRT(B - A )

(C18) /*
```

DERIVATIVE AND ANTIDERIVATIVE EXAMPLES:

```
Enter the expression: */

(X+4)/((X+3)*(X+2)*(X+1));
```

(D18)
$$\frac{X + 4}{(X + 1)\ (X + 2)\ (X + 3)}$$

```
(C19) /* Differentiate this expression with respect to X and simplify the result. */

FACTOR(RATSIMP(DIFF(%,X)));
```

(D19)
$$- \frac{2\ (X^3 + 9\ X^2 + 24\ X + 19)}{(X + 1)^2\ (X + 2)^2\ (X + 3)^2}$$

```
(C20) /* Now, we will integrate this result with respect to X to obtain the antiderivative, or the original expression. */

FACTOR(INTEGRATE(%,X));
```

(D20)
$$\frac{X + 4}{(X + 1)\ (X + 2)\ (X + 3)}$$

```
(C21) /* Let us next compute the 4th derivative of this expression and simplify the result. */

FACTOR(RATSIMP(DIFF(%,X,4)));
```

(D21)
$$\frac{24\ (5\ X^9 + 120\ X^8 + 1210\ X^7 + 6850\ X^6 + 24241\ X^5 + 55990\ X^4 + 84840\ X^3 + 81670\ X^2 + 45480\ X + 11194)}{(X + 1)^5\ (X + 2)^5\ (X + 3)^5}$$

```
(C22) /* Now, let's evaluate this 4th derivative expression at the point X=0 to obtain a numerical result in rational form. */

EV(%,X=0);
```

(D22)
$$\frac{5597}{162}$$

```
(C23) /* Let us then convert this fraction to a decimal number. */

EV(%,NUMER);
(D23)                                34.5493827
```

```
(C24) /* Finally, we will successively integrate each expression beginning with the 4th derivative expression until we obtain
the original expression. */

FACTOR(INTEGRATE(%TH(3),X));
```

(D24)
$$- \frac{12\ (2\ X^7 + 38\ X^6 + 290\ X^5 + 1175\ X^4 + 2760\ X^3 + 3788\ X^2 + 2832\ X + 895)}{(X + 1)^4\ (X + 2)^4\ (X + 3)^4}$$

```
(C25) /*
```

```
8/
FACTOR(INTEGRATE(%,X))$

                 5       4        3        2
           2 (3 X  + 42 X  + 217 X  + 528 X  + 612 X + 274)
(D25)      ------------------------------------------------
                            3        3        3
                     (X + 1)  (X + 2)  (X + 3)

(C26) FACTOR(INTEGRATE(%,X))$
                     3      2
                 2 (X  + 9 X  + 24 X + 19)
(D26)          - ---------------------------
                        2        2        2
                 (X + 1)  (X + 2)  (X + 3)

(C27) FACTOR(INTEGRATE(%,X))$
                          X + 4
(D27)            -----------------------
                 (X + 1) (X + 2) (X + 3)

(C28) /*
```

```
(C1) BATCH(DRI5,1,DSK,DRINK);

(C2) /*



DEFINITE INTEGRAL EXAMPLES;


*/

ALLOC(4);
(D2)                                                      DONE

(C3) DYNAMALLOC:TRUE;
(D3)                                                      TRUE

(C4) /* The next two examples demonstrate the evaluation of definite integrals of real functions of the double and triple
iterated form.  In the first example, MACSYMA asks the user, when attempting to evaluate the integral expression, whether
the variable Y is ZERO or NONZERO.  Normally, in an interactive mode, the user decides what the correct response should be
and responds by typing in 'ZERO;' or 'NONZERO;'.  MACSYMA then continues the evaluation process.  However, when the user
anticipates the sort of question MACSYMA will ask, the user may make these assumptions in advance of the actual
evaluation and later cancel them when no longer appropriate.  This allows for the continuous uninterrupted batching-in
of a file without the system sitting idle waiting for the user to respond to the query. */

ASSUME(Y > 0);
(D4)                                                    [Y > 0]

(C5) /* We first define a function F(X): */

F(X):=X^2;
                                                              2
(D5)                                                  F(X) := X

(C6) /* Express the double iterated integral: */

'INTEGRATE('INTEGRATE(F(X),X,0,SQRT(Y)),Y,0,3);
                                                3  SQRT(Y)
                                               /  /
                                               [  [         2
(D6)                                           I  I        X  dX dY
                                               ]  ]
                                              /  /
                                               0  0

(C7) /*
```

```
Evaluate the integral expression: */

EV(%,INTEGRATE);

DEFINT FASL DSK MACSYM being loaded
Loading done

LIMIT FASL DSK MACSYM being loaded
Loading done

RESIDU FASL DSK MACSYM being loaded
Loading done

RPART FASL DSK MACSYM being loaded
Loading done

SIN FASL DSK MACSYM being loaded
Loading done

SININT FASL DSK MACSYM being loaded
Loading done

SCHATC FASL DSK MACSYM being loaded
Loading done
                                                        6 SQRT(3)
(D7)                                                    ---------
                                                            5

(C8) /* Convert this fractional result to a decimal number: */

EV(%,NUMER);
(D8)                                                    2.07846096

(C9) /* Here, we are making a further assumption that X in the next example is positive: */

ASSUME(X > 0);
(D9)                                                    [X > 0]

(C10) /* Then, we define a new function G(Z): */

G(Z):=(1+Z^(1/3))/SQRT(Z);
                                                                1/3
                                                           1 + Z
(D10)                                             G(Z) := --------
                                                          SQRT(Z)

(C11) /* Now, we are going to set up and express the triple iterated integral: */

'INTEGRATE('INTEGRATE('INTEGRATE(G(Z),Z,0,Y),Y,0,X),X,0,1);
                                                 1  X  Y
                                                 /  /  /   1/3
                                                 [  [  [  Z    + 1
(D11)                                            I  I  I  -------- dZ dY dX
                                                 ]  ]  ]  SQRT(Z)
                                                 /  /  /
                                                 0  0  0

(C12) /*
```

```
Then evaluate the integral expression: */

ASSUME(Y>0)$
(D12)                                                   [Y > 0]

(C13) %TH(2)$
                                            1   X   Y
                                            /   /   /   1/3
                                            [   [   [  Z     + 1
(D13)                                       I   I   I  --------- dZ dY dX
                                            ]   ]   ]  SQRT(Z)
                                            /   /   /
                                            0   0   0

(C14) EV(%,INTEGRATE)$

SOLVE FASL DSK MACSYM being loaded
Loading done
                                                          2144
(D14)                                                     ----
                                                          2805

(C15) /* Convert the rational form of the answer to a decimal number: */

EV(%,NUMER)$
(D15)                                                  0.76434938

(D16)                                                  BATCH DONE

(C17) ^Z
[DDT]
*
```

```
VECTOR CALCULUS EXAMPLES:

At this point we would like to demonstrate one of the special set of routines in MACSYMA for handling vector differential
calculus operations.  We must first load these routines into core.  The comments following the sentence which begins with 'Note:'
belong to the file being batched in at this time.  The terminal printout was turned off during the batching-in
process thus eliminating the printing of all the code for the vector routines.  The command which precedes the BATCH command
turns the terminal printout Off.  Note:  The current version of VECT is the one due to Stoutemyer.
                                     It will be replaced soon by an extended version which handles both
                                     vectors and dyadics.
                                             Michael C. Wirth (MCW)
                                             12/18/79
*/

TTYOFF:TRUE;

(D100)                                                  BATCH DONE

(C101) /* We will begin by defining two 3-dimensional vector functions A(X,Y,Z) and B(X,Y,Z).  MACSYMA represents these vector
functions as a list of the coefficients of the implied unit vectors i,j,&k. */

A(X,Y,Z):=[X^2*Z,-2*Y^3*Z^2,X*Y^2*Z];
                                                         2            3  2      2
(D101)                                        A(X, Y, Z) := [X  Z, - 2 Y  Z , X Y  Z]

(C102) B(X,Y,Z):=[X*Y*Z,3*X^2*Y,(X*Z^2-Y^2*Z)];
                                                                  2         2    2
(D102)                                       B(X, Y, Z) := [X Y Z, 3 X  Y, X Z  - Y  Z]

(C103) /* This next command turns on all the various expansion flags used by VECT.  These can be set individually or all at
one time as is the case here. */

EXPANDALL:TRUE;
(D103)                                                    TRUE

(C104) /* Now, let's express the cross product of these two vector functions.  The vector cross product operator is "~". */

EXPRESS(A(X,Y,Z)~B(X,Y,Z));
                    3  2      2    2          3  3      2  3  2     2       2    2          4  3      4
(D104)         [- 2 Y  Z  (X Z  - Y  Z) - 3 X  Y  Z, X  Y  Z  - X  Z (X Z  - Y  Z), 2 X Y  Z  + 3 X  Y Z]

(C105) /* Define a vector function C(X,Y,Z) which is the result of A(X,Y,Z)~B(X,Y,Z): */

C(X,Y,Z):=[-2*Y^3*Z^2*(X*Z^2-Y^2*Z)-3*X^3*Y^3*Z,X^2*Y^3*Z^2-X^2*Z*(X*Z^2-Y^2*Z),2*X*Y^4*Z^3+3*X^4*Y*Z];
                           3  2      2    2          3  3      2  3  2     2       2    2          4  3      4
(D105)     C(X, Y, Z) := [- 2 Y  Z  (X Z  - Y  Z) - 3 X  Y  Z, X  Y  Z  - X  Z (X Z  - Y  Z), 2 X Y  Z  + 3 X  Y Z]

(C106) /*
```

```
First, we will express the divergence of this vector function, then compute the derivatives in the expression,
and finally we will evaluate the expression at the point (X,Y,Z)=(1,-1,1): */

EXPRESS(DIV(C(X,Y,Z)));
              d        4  3        4             d          3  2      2    2          3  3            d     2  3  2     2      2    2
(D106)        -- (2 X Y  Z  + 3 X  Y Z) + -- (- 2 Y  Z  (X Z  - Y  Z) - 3 X  Y  Z) + -- (X  Y  Z  - X  Z (X Z  - Y  Z))
              dZ                             dX                                      dY

(C107) %,DIFF;
                                     3  4        4  2        2  2  2      2    2      2  3          4
(D107)                          - 2 Y  Z  + 6 X Y  Z  + 3 X  Y  Z  + 2 X  Y Z  - 9 X  Y  Z + 3 X  Y

(C108) EV(%,X=1,Y=-1,Z=1);
(D108)                                                        15

(C109) /* Next, we express the curl of the vector function C(X,Y,Z) and compute the derivatives of the expression: */

EXPRESS(CURL(C(X,Y,Z)));
         d        4  3        4         d     2  3  2     2      2    2
(D109) [-- (2 X Y  Z  + 3 X  Y Z) - -- (X  Y  Z  - X  Z (X Z  - Y  Z)),
        dY                            dZ

d          3  2      2    2          3  3      d        4  3        4
-- (- 2 Y  Z  (X Z  - Y  Z) - 3 X  Y  Z) - -- (2 X Y  Z  + 3 X  Y Z),
dZ                                           dX

d     2  3  2     2      2    2        d          3  2      2    2          3  3
-- (X  Y  Z  - X  Z (X Z  - Y  Z)) - -- (- 2 Y  Z  (X Z  - Y  Z) - 3 X  Y  Z)]
dX                                     dY

(C110) %,DIFF;
              3  3      2      2    2       2                 2       2  3          4
(D110) [8 X Y  Z  + X  (X Z  - Y  Z) + X  Z (2 X Z - Y ) - 2 X  Y  Z + 3 X  Z,

     4  3      3      2    2        3  2              2        3             3  3
- 2 Y  Z  - 4 Y  Z (X Z  - Y  Z) - 2 Y  Z  (2 X Z - Y ) - 12 X  Y Z - 3 X  Y ,

     4  3    2  3        2  2     2    2                2    2            3  2        3  2
- 4 Y  Z  - X  Z  + 6 Y  Z  (X Z  - Y  Z) - 2 X Z (X Z  - Y  Z) + 2 X Y  Z  + 9 X  Y  Z]

(C111) /* Now, we will compute the vector potential of this vector function: */

VECTORPOTENTIAL(%);
            4        3    4        5  3         3    2        3  3         3  4      3  3            2  3        2  2       4    2
          (Y  + 4 X Y ) Z  - 4 Y  Z  + 12 X  Y Z  + 6 X  Y  Z   4 X Y  Z  + 2 X  Z  + (- 2 X  Y  - 2 X  Y  + 3 X ) Z
(D111) [- ----------------------------------------------------, - ----------------------------------------------------, 0]
                                   2                                                     2

(C112) /* Next, we define the following scalar function F(X,Y,Z) as the dot product of the vector functions A(X,Y,Z)
and B(X,Y,Z): */

EXPRESS(A(X,Y,Z).B(X,Y,Z));
                                       2       2    2        2  4  2    3    2
(D112)                            X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z

(C113) /*
```

```
*/

F(X,Y,Z):=X*Y^2*Z*(X*Z^2-Y^2*Z)-6*X^2*Y^4*Z^2+X^3*Y*Z^2;
                                   2        2     2            2  4  2     3     2
(D113)                   F(X, Y, Z) := X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z

(C114) /* We express the gradient of this scalar function and compute the derivatives of the expression: */

EXPRESS(GRAD(F(X,Y,Z)));
        d       2         2     2          2  4  2     3     2     d       2         2     2          2  4  2     3     2
(D114) [-- (X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z ), -- (X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z ),
        dX                                                dY

                                                                          d       2         2     2          2  4  2     3     2
                                                                          -- (X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z )]
                                                                          dZ

(C115) %,DIFF;
            2  3     2         2     2             4  2       2     2              2     2           2  3  2          3  2     3  2
(D115) [X Y  Z  + Y  Z (X Z  - Y  Z) - 12 X Y  Z  + 3 X  Y Z , 2 X Y Z (X Z  - Y  Z) - 24 X  Y  Z  - 2 X Y  Z  + X  Z ,

                                                              2     2     2           2               2         2  4         3
                                                           X Y  (X Z  - Y  Z) + X Y  Z (2 X Z - Y ) - 12 X  Y  Z + 2 X  Y Z]

(C116) /* Next, we express the divergence of this gradient and compute the derivatives of this expression: */

EXPRESS(DIV(GRAD(F(X,Y,Z))));
        2                                                      2
       d       2         2     2          2  4  2     3     2     d       2         2     2          2  4  2     3     2
(D116) --- (X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z ) + --- (X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z )
         2                                                  2
       dZ                                                  dY

                                                                         2
                                                                        d       2         2     2          2  4  2     3     2
                                                                      + --- (X Y  Z (X Z  - Y  Z) - 6 X  Y  Z  + X  Y Z )
                                                                          2
                                                                        dX

(C117) %,DIFF;
          2  3               2     2          4  2          2  2  2          2  2             2         2                2         2  2          2  4
(D117) 2 Y  Z  + 2 X Z (X Z  - Y  Z) - 12 Y  Z  - 72 X  Y  Z  - 10 X Y  Z  + 6 X Y Z  + 2 X Y  (2 X Z - Y ) + 2 X  Y  Z - 12 X  Y

                                                                                                                           3
                                                                                                                     + 2 X  Y

(C118) /* Finally, we can expand this expression: */

EXPAND(%);
            2  3       2  3        4  2        2  2  2        2  2         2       2  2         2  4          4      3
(D118)     2 Y  Z  + 2 X  Z  - 12 Y  Z  - 72 X  Y  Z  - 12 X Y  Z  + 6 X Y Z  + 6 X  Y  Z - 12 X  Y  - 2 X Y  + 2 X  Y

(D119)                                        BATCH DONE

(C120) ^Z
[DDT]
*
```

```
(C1) BATCH(DRI6,1,DSK,DRINK);

(C2) /*

INTEGRATION BY SPECIAL CONTOURS AND RESIDUE THEOREM EXAMPLES:


The following examples demonstrate the evaluation of definite integrals of real valued functions by the use of
1) special contours and 2) the residue theorem.  Examples are from ref.[3].

First, we will work with definite integrals involving trigonometric functions: */

ALLOC(4);
(D2)                                                               DONE

(C3) DYNAMALLOC:TRUE;
(D3)                                                               TRUE

(C4) 'INTEGRATE(COS(X)/(5+4*COS(X)),X,-%PI,%PI);
                                                        %PI
                                                       /
                                                       [          COS(X)
(D4)                                                   I       ------------ dX
                                                       ]       4 COS(X) + 5
                                                       /
                                                        - %PI

(C5) EV(%,INTEGRATE);

DEFINT FASL DSK MACSYM being loaded
Loading done

LIMIT FASL DSK MACSYM being loaded
Loading done

RESIDU FASL DSK MACSYM being loaded
Loading done

RPART FASL DSK MACSYM being loaded
Loading done

SOLVE FASL DSK MACSYM being loaded
Loading done

ATRIG FASL DSK MAXOUT being loaded
Loading done

LOGARC FASL DSK MACSYM being loaded
Loading done

ASKP FASL DSK MACSYM being loaded
Loading done

HAYAT FASL DSK MACSYM being loaded
Loading done
                                                                  %PI
(D5)                                                            - ---
                                                                   3

(C6) /*
```

```
In the next example we assume that N is positive and nonzero. */

ASSUME(N>0);
(D6)                                                     [N > 0]

(C7) 'INTEGRATE(SIN(X)^(2*N),X,0,%PI);
                                                       %PI
                                                      /
                                                      [           2 N
(D7)                                                  I    SIN(X)     dX
                                                      ]
                                                      /
                                                       0

(C8) EV(%,INTEGRATE);

GAMMA FASL DSK MAXOUT being loaded
Loading done
                                                          2 N + 1  1
(D8)                                                 BETA(-------, -)
                                                             2     2

(C9) /*

We evaluate the Beta function for N=5! */

EV(%,N=5);

BINOML FASL DSK MAXOUT being loaded
Loading done
                                                          63 %PI
(D9)                                                      ------
                                                           256

(C10) /* We are through with our assumption about N. */

FORGET(N>0);
(D10)                                                    [N > 0]

(C11) /* The next few examples deal with improper integrals involving trigonometric functions.  In this next example we will
assume that A is greater than zero.  Also, note that MINF is minus infinity and INF is plus infinity. */

ASSUME(A > 0);
(D11)                                                    [A > 0]

(C12) 'INTEGRATE(X*SIN(A*X)/(X^4+4),X,MINF,INF);
                                                      INF
                                                     /
                                                     [      X SIN(A X)
(D12)                                                I      ---------- dX
                                                     ]        4
                                                     /       X  + 4
                                                      MINF

(C13) EV(%,INTEGRATE);
                                                             - A
                                                      %PI %E     SIN(A)
(D13)                                                 ----------------
                                                             2

(D14)                                                    BATCH DONE
```

```
(C1) DEMO(DRI7,1,DSK,DRINK);

(C2) /* Note that this file is being DEMOed in rather than being BATCHed in.  The little '_' marks are pauses, and the
space bar must be used in order to move on to the next command. */

DYNAMALLOC:TRUE;
(D2)                                                          TRUE
_
(C3) ASSUME(B > 0);
(D3)                                                        [B > 0]
_
(C4) 'INTEGRATE(%E^-X^2*COS(2*B*X),X,0,INF);
                                                  INF
                                                 /         2
                                                 [      - X
(D4)                                             I    %E      COS(2 B X) dX
                                                 ]
                                                 /
                                                  0
_
(C5) EV(%,INTEGRATE);

DEFINT FASL DSK MACSYM being loaded
Loading done

LIMIT FASL DSK MACSYM being loaded
Loading done

RESIDU FASL DSK MACSYM being loaded
Loading done

RPART FASL DSK MACSYM being loaded
Loading done

ASKP FASL DSK MACSYM being loaded
Loading done

SIN FASL DSK MACSYM being loaded
Loading done

SININT FASL DSK MACSYM being loaded
Loading done

SCHATC FASL DSK MACSYM being loaded
Loading done

RISCH FASL DSK MACSYM being loaded
Loading done

PFRAC FASL DSK MAXOUT being loaded
Loading done

ERF FASL DSK MAXOUT being loaded
Loading done
                                                                  2
                                                               - B
                                                    SQRT(%PI) %E
(D5)                                                ----------------
                                                           2
_
```

```
(C6) FORGET(B > 0);
(D6)                                               [B > 0]

(C7) 'INTEGRATE(SIN(X)/X,X,0,INF);

                                                  INF
                                                 /
                                                 [     SIN(X)
(D7)                                             I     ------ dX
                                                 ]       X
                                                 /
                                                  0

(C8) EV(%,INTEGRATE);

SOLVE FASL DSK MACSYM being loaded
Loading done
                                                      %PI
(D8)                                                  ---
                                                       2

_^Z
[DDT]
*
```

```
(C1) DEMO(DRI7A,1,DSK,DRINK);

(C2) DYNAMALLOC:TRUE;
(D2)                                                  TRUE

(C3) ASSUME(A > 0,B > 0);
(D3)                                             [A > 0, B > 0]

(C4) 'INTEGRATE(COS(A*X)/(X^2+B^2)^2,X,0,INF);
                                                INF
                                               /
                                               [      COS(A X)
(D4)                                           I     ---------- dX
                                               ]       2    2 2
                                               /     (X  + B )
                                                0

(C5) EV(%,INTEGRATE);

DEFINT FASL DSK MACSYM being loaded
Loading done

LIMIT FASL DSK MACSYM being loaded
Loading done

RESIDU FASL DSK MACSYM being loaded
Loading done

RPART FASL DSK MACSYM being loaded
Loading done

SOLVE FASL DSK MACSYM being loaded
Loading done

ASKP FASL DSK MACSYM being loaded
Loading done

HAYAT FASL DSK MACSYM being loaded
Loading done
                                                                 - A B
                                           (%PI A B + %PI) %E
(D5)                                       -----------------------
                                                         3
                                                      4 B

(C6) FORGET(A>0,B>0);
(D6)                                             [A > 0, B > 0]
```

$$\int_0^{\infty} \frac{\cos(A X)}{(X^2 + B^2)^2}\, dX = \frac{(\pi A B + \pi)\, e^{-A B}}{4 B^3}$$

```
(C7) ASSUME(A>B,B>0);
(D7)                                                  [A > B, B > 0]

(C8) 'INTEGRATE(COS(X)/((X^2+A^2)*(X^2+B^2)),X,MINF,INF);
                                            INF
                                           /
                                           [            COS(X)
(D8)                                       I   ------------------------ dX
                                           ]     2    2     2    2
                                           /   (X  + A ) (X  + B )
                                            MINF

(C9) EV(%,INTEGRATE);

                                          - B           B            A
                                        %E    (%PI B %E  - %PI A %E )
(D9)                                    ---------------------------------
                                                A  3    3   A
                                           A %E  B  - A  %E  B

(D10)                                            BATCH DONE

(C11) ^Z
[DDT]
*
```

2.3 Matrix Operations

The matrix manipulation functions in MACSYMA are excellent in the sense that they are easy to use and relatively fast. Not all the matrix manipulation features are demonstrated here, but many of the more important features are shown.* It should be re-emphasized that reading thoroughly Demonstration 2.1 on "Simplification of Algebraic Expressions" is important since the results of many matrix operations .require simplification of rational coefficients in order to transform them into a useful form. The example of this section dealing with the Vandermonde matrix illustrates this.

MACSYMA has an excellent EIGEN package in the SHARE directory. Its purpose is to compute symbolically eigenvectors and eigenvalues of matrices. MACSYMA can also be used to manipulate Jacobians of transformations and Wronskians; however, in the interest of brevity, we do not include examples of these features here.

One important restriction concerning symbolic matrix manipulation limitations in MACSYMA needs to be mentioned. If one is inverting a matrix composed of symbolic elements, as opposed to numerical elements, the maximum matrix size may be limited. For example, it was found that the maximum Vandermonde matrix that could be successfully inverted using MACSYMA, with maximum core allocated, was a 5×5. The reason for this is that "intermediate expression swell" due to the generation of large rational expressions during the inversion computation causes the exhaustion of the user's list space for Vandermonde matrices of size larger than 5. On the other hand, matrices containing up to 20×20 numerical elements have been successfully inverted symbolically using MACSYMA.

*The examples for this demonstration were suggested by Marvin J. Goldstein.

```
(C1) BATCH(DRIB,1,DSK,DRINK);

(C2) /*

                         DEMONSTRATION 2.3 - MATRIX OPERATIONS


Let us first define an array function which we can use to define and generate a Hilbert matrix! */

ARRAY(H,10,10);
(D2)                                               H

(C3) H[I,K]:=1/(I+K-1);
                                                                1
(D3)                                           H     := ---------
                                                I, K     I + K - 1

(C4) /* Now, we will generate a Hilbert matrix (which in this particular case also turns out to be a Hankel matrix) using
this array function! */

MM:GENMATRIX(H,6,6);

GENMAT FASL DSK MAXOUT being loaded
Loading done
                                          [    1  1  1  1   1  ]
                                          [ 1  -  -  -  -   -  ]
                                          [    2  3  4  5   6  ]
                                          [                    ]
                                          [ 1  1  1  1  1   1  ]
                                          [ -  -  -  -  -   -  ]
                                          [ 2  3  4  5  6   7  ]
                                          [                    ]
                                          [ 1  1  1  1  1   1  ]
                                          [ -  -  -  -  -   -  ]
                                          [ 3  4  5  6  7   8  ]
(D4)                                      [                    ]
                                          [ 1  1  1  1  1   1  ]
                                          [ -  -  -  -  -   -  ]
                                          [ 4  5  6  7  8   9  ]
                                          [                    ]
                                          [ 1  1  1  1  1   1  ]
                                          [ -  -  -  -  -   -- ]
                                          [ 5  6  7  8  9   10 ]
                                          [                    ]
                                          [ 1  1  1  1  1   1  ]
                                          [ -  -  -  -  --  -- ]
                                          [ 6  7  8  9  10  11 ]

(C5) /*
```

```
Then, we will compute its inverse.  This is a symbolic (non-numerical) computation so that the inverse is exact
(See ref.[4]: Computer Solution Of Linear Algebraic Systems by G. Forsythe and C. B. Moler, P-H, page 82(19.5) for verification).
Note the use of the double "^^" operator for computing matrix inverses in MACSYMA. */

HHINV:HH^^(-1);

MDOT FASL DSK MACSYM being loaded
Loading done
                                   [   36     - 630      3360     - 7560      7560     - 2772  ]
                                   [                                                           ]
                                   [ - 630    14700    - 88200    211680   - 220500    83160   ]
                                   [                                                           ]
                                   [  3360   - 88200    564480   - 1411200  1512000  - 582120  ]
(D5)                               [                                                           ]
                                   [ - 7560   211680  - 1411200  3628800  - 3969000   1552320  ]
                                   [                                                           ]
                                   [  7560  - 220500   1512000  - 3969000  4410000  - 1746360  ]
                                   [                                                           ]
                                   [ - 2772   83160   - 582120   1552320  - 1746360   698544   ]

(C6) /*  We are now going to define an array function which we will use to define and generate an orthogonal elementary
permutation (or transformation) matrix which we will call an E matrix: */

ARRAY(J,10,10);
(D6)                                                        J

(C7) N:6;
(D7)                                                        6

(C8) /* We are going to use an iterative looping statement similar to those used in FORTRAN, ALGOL, PASCAL, etc. */

FOR I THRU 6 DO (FOR K THRU 6 DO (IF I+K = N+1 THEN J[I,K]:1 ELSE J[I,K]:0));
(D8)                                                      DONE

(C9) /* Here, we generate the elementary matrix E: */

E:GENMATRIX(J,N,N);
                                                  [ 0  0  0  0  0  1 ]
                                                  [                  ]
                                                  [ 0  0  0  0  1  0 ]
                                                  [                  ]
                                                  [ 0  0  0  1  0  0 ]
(D9)                                              [                  ]
                                                  [ 0  0  1  0  0  0 ]
                                                  [                  ]
                                                  [ 0  1  0  0  0  0 ]
                                                  [                  ]
                                                  [ 1  0  0  0  0  0 ]

(C10) /*
```

```
Post-multiplying the Hilbert matrix with the E (elementary) matrix interchanges the columns of the Hilbert matrix
thus yielding a Toeplitz matrix.  Note that matrix multiplication requires the noncommutative operator '.'. */

T:HH . E;
                                        [ 1    1    1   1   1      ]
                                        [ -    -    -   -   -   1  ]
                                        [ 6    5    4   3   2      ]
                                        [                          ]
                                        [ 1    1    1   1   1   1  ]
                                        [ -    -    -   -   -   -  ]
                                        [ 7    6    5   4   3   2  ]
                                        [                          ]
                                        [ 1    1    1   1   1   1  ]
                                        [ -    -    -   -   -   -  ]
                                        [ 8    7    6   5   4   3  ]
(D10)                                   [                          ]
                                        [ 1    1    1   1   1   1  ]
                                        [ -    -    -   -   -   -  ]
                                        [ 9    8    7   6   5   4  ]
                                        [                          ]
                                        [ 1    1    1   1   1   1  ]
                                        [ --   -    -   -   -   -  ]
                                        [ 10   9    8   7   6   5  ]
                                        [                          ]
                                        [ 1    1    1   1   1   1  ]
                                        [ --   --   -   -   -   -  ]
                                        [ 11   10   9   8   7   6  ]

(C11) /* Here, we compute the inverse of the Toeplitz matrix: */

TINV:T^^(-1);
                  [ - 2772    83160    - 582120   1552320   - 1746360   698544   ]
                  [                                                              ]
                  [  7560   - 220500   1512000   - 3969000   4410000   - 1746360 ]
                  [                                                              ]
                  [ - 7560    211680  - 1411200   3628800   - 3969000   1552320  ]
                  [                                                              ]
(D11)             [  3360   - 88200    564480    - 1411200   1512000   - 582120  ]
                  [                                                              ]
                  [ - 630     14700    - 88200    211680    - 220500    83160    ]
                  [                                                              ]
                  [  36     - 630      3360      - 7560      7560      - 2772    ]

(C12) /* Pre-multiplying the inverse of the Toeplitz matrix with the E matrix interchanges the rows of the Toeplitz inverse thus
giving us the inverse of the Hilbert matrix: */

E . TINV;
                  [  36     - 630      3360      - 7560      7560      - 2772    ]
                  [                                                              ]
                  [ - 630     14700    - 88200    211680    - 220500    83160    ]
                  [                                                              ]
                  [  3360   - 88200    564480    - 1411200   1512000   - 582120  ]
                  [                                                              ]
(D12)             [ - 7560    211680  - 1411200   3628800   - 3969000   1552320  ]
                  [                                                              ]
                  [  7560   - 220500   1512000   - 3969000   4410000   - 1746360 ]
                  [                                                              ]
                  [ - 2772    83160    - 582120   1552320   - 1746360   698544   ]

(C13) /*
```

```
The rank of the Toeplitz matrix is: */

RANK(T);
(D13)                                          6

(C14) /* The rank of the Toeplitz inverse is: */

RANK(TINV);
(D14)                                          6

(C15) /* The determinant of the Hilbert matrix is: */

DETERMINANT(HH);
                                               1
(D15)                                ------------------
                                     186313420339200000

(C16) /* The determinant of the Toeplitz inverse is: */

DETERMINANT(TINV);
(D16)                                - 186313420339200000

(C17) /* Now, we are going to generate a (4x4) Vandermonde matrix by explicitly stating the elements, separated by commas, of
each row enclosed in square brackets separated by commas. */

VANDER:MATRIX([1,A,A^2,A^3],[1,B,B^2,B^3],[1,C,C^2,C^3],[1,D,D^2,D^3]);
                                               [          2   3 ]
                                               [ 1   A   A   A  ]
                                               [                ]
                                               [          2   3 ]
                                               [ 1   B   B   B  ]
(D17)                                          [                ]
                                               [          2   3 ]
                                               [ 1   C   C   C  ]
                                               [                ]
                                               [          2   3 ]
                                               [ 1   D   D   D  ]

(C18) /* Let's multiply the Vandermonde matrix by itself: */

VANDER . VANDER;
                [  3    2              3     2                 3  2     2  2        2     2     3  3     2  3        3     3 ]
                [ A  + A  + A + 1    A  D + A  C + A B + A    A  D  + A  C  + A B  + A     A  D  + A  C  + A B  + A  ]
                [                                                                                                    ]
                [  3    2              3     2      2          3  2     2  2    3     2     3  3     2  3    4     3 ]
                [ B  + B  + B + 1    B  D + B  C + B  + A     B  D  + B  C  + B  + A      B  D  + B  C  + B  + A     ]
(D18)           [                                                                                                    ]
                [  3    2              3      3                3  2    4      2     2     3  3    5     3     3 ]
                [ C  + C  + C + 1    C  D + C  + B C + A      C  D  + C  + B  C + A       C  D  + C  + B  C + A      ]
                [                                                                                                    ]
                [  3    2              4      2                5      2  2    2     2     6      3  2    3     3 ]
                [ D  + D  + D + 1    D  + C D  + B D + A      D  + C  D  + B  D + A       D  + C  D  + B  D + A      ]

(C19) /*
```

```
The determinant of the Vandermonde matrix is: */

DETERMINANT(VANDER);
                 2   3    3     2 3    3   2    2     3 2     2       3    3       3   3          3         2 3     3 2
(D19) - A (- B  (D  - C ) + C  D  + B  (D  - C ) - C  D ) + A  (- B (D  - C ) + C D  + B  (D - C) - C  D) + B (C  D  - C  D )

                                              2     3    3      3       2    2        2    2           2        3     2    2
                                          - B  (C D  - C  D) - A  (- B (D  - C ) + C D  + B  (D - C) - C  D) + B  (C D  - C  D)

(C20) /* We factor to simplify the determinant expression: */

FACTOR(%);
(D20)                        (B - A) (C - A) (C - B) (D - A) (D - B) (D - C)

(C21) /* The rank of the Vandermonde matrix is: */

RANK(VANDER);
(D21)                                              4

(C22) /* Next, we will compute the inverse of the Vandermonde matrix in factored form: */

VANDERINV:FACTOR(VANDER^^(-1));
        [         B C D                        A C D                        A B D                          A B C         ]
        [ -----------------------  - -----------------------  -----------------------  - -----------------------  ]
        [ (B - A) (C - A) (D - A)    (B - A) (C - B) (D - B)  (C - A) (C - B) (D - C)    (D - A) (D - B) (D - C)  ]
        [                                                                                                               ]
        [     C D + B D + B C            C D + A D + A C           B D + A D + A B              B C + A C + A B       ]
        [ - -----------------------  -----------------------  - -----------------------  -----------------------  ]
        [   (B - A) (C - A) (D - A)  (B - A) (C - B) (D - B)    (C - A) (C - B) (D - C)  (D - A) (D - B) (D - C)  ]
(D22)   [                                                                                                               ]
        [        D + C + B                  D + C + A                  D + B + A                    C + B + A         ]
        [ -----------------------  - -----------------------  -----------------------  - -----------------------  ]
        [ (B - A) (C - A) (D - A)    (B - A) (C - B) (D - B)  (C - A) (C - B) (D - C)    (D - A) (D - B) (D - C)  ]
        [                                                                                                               ]
        [            1                          1                          1                            1             ]
        [ - -----------------------  -----------------------  - -----------------------  -----------------------  ]
        [   (B - A) (C - A) (D - A)  (B - A) (C - B) (D - B)    (C - A) (C - B) (D - C)  (D - A) (D - B) (D - C)  ]

(C23) /* We will now multiply the Vandermonde matrix by its inverse to yield the identity matrix.  Because MACSYMA
doesn't simplify the rational expressions unless asked to by the user, it won't be immediately apparent that this
is the identity matrix.  Since the expressions in the matrix being generated are too large for the entire matrix
expression to fit on a page, we first do the computation without any display (by terminating the input command with a "$"
instead of a ";"), and then we display the identity matrix one column at a time prior to simplification. */

ID:COMBINE(VANDER . VANDERINV)$

COMBF FASL DSK MAXOUT being loaded
Loading done

(C24) /*
```

```
1st column: */
COL(ID,1);
```

```
      [                            2                      3 ]
      [ - A (C D + B D + B C) + A  (D + C + B) + B C D - A  ]
      [ --------------------------------------------------- ]
      [              (B - A) (C - A) (D - A)                ]
      [                                                     ]
      [                            2                      3 ]
      [ - B (C D + B D + B C) + B  (D + C + B) + B C D - B  ]
      [ --------------------------------------------------- ]
      [              (B - A) (C - A) (D - A)                ]
(D24) [                                                     ]
      [                            2                      3 ]
      [ - C (C D + B D + B C) + C  (D + C + B) + B C D - C  ]
      [ --------------------------------------------------- ]
      [              (B - A) (C - A) (D - A)                ]
      [                                                     ]
      [    3                            2                   ]
      [ - D  - D (C D + B D + B C) + D  (D + C + B) + B C D ]
      [ --------------------------------------------------- ]
      [              (B - A) (C - A) (D - A)                ]
```

```
(C25) /* 2nd column: */
COL(ID,2);
```

```
      [                          2                      3 ]
      [ A (C D + A D + A C) - A  (D + C + A) - A C D + A  ]
      [ ------------------------------------------------- ]
      [            (B - A) (C - B) (D - B)                ]
      [                                                   ]
      [                          2                      3 ]
      [ B (C D + A D + A C) - B  (D + C + A) - A C D + B  ]
      [ ------------------------------------------------- ]
      [            (B - A) (C - B) (D - B)                ]
(D25) [                                                   ]
      [                          2                      3 ]
      [ C (C D + A D + A C) - C  (D + C + A) - A C D + C  ]
      [ ------------------------------------------------- ]
      [            (B - A) (C - B) (D - B)                ]
      [                                                   ]
      [  3                            2                   ]
      [ D  + D (C D + A D + A C) - D  (D + C + A) - A C D ]
      [ ------------------------------------------------- ]
      [            (B - A) (C - B) (D - B)                ]
```

```
(C26) /*
```

3rd column: */

COL(ID,3);

(D26)

$$\begin{bmatrix} \dfrac{-A(BD+AD+AB)+A^2(D+B+A)+ABD-A^3}{(C-A)(C-B)(D-C)} \\[2ex] \dfrac{-B(BD+AD+AB)+B^2(D+B+A)+ABD-B^3}{(C-A)(C-B)(D-C)} \\[2ex] \dfrac{-C(BD+AD+AB)+C^2(D+B+A)+ABD-C^3}{(C-A)(C-B)(D-C)} \\[2ex] \dfrac{-D^3-D(BD+AD+AB)+D^2(D+B+A)+ABD}{(C-A)(C-B)(D-C)} \end{bmatrix}$$

(C27) /* 4th column: */

COL(ID,4);

(D27)

$$\begin{bmatrix} \dfrac{A(BC+AC+AB)-A^2(C+B+A)-ABC+A^3}{(D-A)(D-B)(D-C)} \\[2ex] \dfrac{B(BC+AC+AB)-B^2(C+B+A)-ABC+B^3}{(D-A)(D-B)(D-C)} \\[2ex] \dfrac{C^3+C(BC+AC+AB)-C^2(C+B+A)-ABC}{(D-A)(D-B)(D-C)} \\[2ex] \dfrac{D^3-(C+B+A)D^2+(BC+AC+AB)D-ABC}{(D-A)(D-B)(D-C)} \end{bmatrix}$$

(C28) /* Now, we will use RATSIMP on this matrix, and we see that it is indeed the identity matrix: */

RATSIMP(ID);

(D28)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(C29) /*

```
This final example demonstrates the manipulation of a matrix whose elements are complex and the use of an interactive
method of entering a matrix in MACSYMA.  This means that although the current demonstration file is being BATCHed-in the
system will pause until each element has been entered before continuing with the batch process.  */

A:ENTERMATRIX(3,3);

ENTERM FASL DSK MAXOUT being loaded
Loading done

Is the matrix  1. Diagonal  2. Symmetric  3. Antisymmetric  4. General
Answer 1, 2, 3 or 4
4;

Row 1 Column 1:  1;

Row 1 Column 2:  1+2*%I;

Row 1 Column 3:  2+10*%I;

Row 2 Column 1:  1+%I;

Row 2 Column 2:  3*%I;

Row 2 Column 3:  -5+14*%I;

Row 3 Column 1:  1+%I;

Row 3 Column 2:  5*%I;

Row 3 Column 3:  -8+20*%I;

Matrix entered.
                                                    [   1      2 %I + 1   10 %I + 2 ]
                                                    [                               ]
(D29)                                               [ %I + 1     3 %I     14 %I - 5 ]
                                                    [                               ]
                                                    [ %I + 1     5 %I     20 %I - 8 ]

(C30) /* Next, we compute the symbolic inverse and the determinant of this matrix.  */

AINV:A^^-1;
                                                  [ %I + 10     6 %I - 2    - 2 %I - 3 ]
                                                  [                                    ]
(D30)                                             [ 9 - 3 %I      8 %I      - 2 %I - 3 ]
                                                  [                                    ]
                                                  [ 2 %I - 2  - 2 %I - 1        1      ]

(C31) EXPAND(DETERMINANT(A));
(D31)                                                            1

(D32)                                                       BATCH DONE

(C33) ^Z
[DDT]
*
```

2.4 Solution of Algebraic Equations

The reader who is interested in the symbolic solution of algebraic equations would benefit from reading all the examples in this section. The examples chosen represent some of the classical categories of equations encountered in scientific and engineering *applications*. There are examples of both single equations and systems of equations. The limitations involved in solving large systems of equations are related to those mentioned in Section 2.3 on "Matrix Operations."

```
(C1) BATCH(DRI9,1,DSK,DRINK)$

(C2) /*

                    DEMONSTRATION 2.4 - SOLUTION OF ALGEBRAIC EQUATIONS


Although there are several function commands in MACSYMA used for solving algebraic equations and systems of algebraic equations,
we are going to demonstrate the one which has the most general flexibility.  This function command is called SOLVE, and it has
two forms based on argument list structure:
              1)SOLVE(EXP,VAR) solves the algebraic equation EXP for the variable VAR and returns a list of solution
                equations in VAR (note: if EXP is not an equation, it is assumed to be set equal to zero).
              2)SOLVE([EQ1,...,EQN],[V1,...,VN]) solves a system of simultaneous (linear or  nonlinear) polynomial
                equations by calling LINSOLVE or ALGSYS and returns a list of the solutions in the variables specified.

The first equation is a simple quadratic: */

X^2+5*X+2 = 0;
                                       2
(D2)                                  X  + 5 X + 2 = 0

(C3) SOLVE(%,X);

SOLVE FASL DSK MACSYM being loaded
Loading done
                                 SQRT(17) + 5      SQRT(17) - 5
(D3)                      [X = - ------------, X = ------------]
                                      2                 2

(C4) /* We can convert these rational solution forms to decimal number form.  The use of the MACSYMA functions FIRST, LAST,
and REST are used to extract the individual solutions from the lists of solutions.  A detailed explanation of their proper use
can be found in ref.[1]. */

EV(FIRST(%),NUMER);
(D4)                                  X = - 4.5615528

(C5) LAST(%TH(2))$

(C6) EV(%,NUMER);
(D6)                                 X = - 0.438447177

(C7) DYNAMALLOC:TRUE;
(D7)                                        TRUE

(C8) /*
```

```
The Guldberg and Waage equation from Chemistry: */

K*(A-X)*(B-X) = (C+X)*(D+X);
(D8)                                       K (A - X) (B - X) = (X + C) (X + D)

(C9) SOLVE(%,X);
(D9) [X =

        2  2    2  2                    2                                                           2    2
- SQRT(B  K  + A  K  + A (B (4 K - 2 K ) + 2 D K + 2 C K) + B (2 D K + 2 C K) + C D (4 K - 2) + D  + C ) + B K + A K + D + C
------------------------------------------------------------------------------------------------------------------------------,
                                                              2 K - 2

          2  2    2  2                    2                                                           2    2
    SQRT(B  K  + A  K  + A (B (4 K - 2 K ) + 2 D K + 2 C K) + B (2 D K + 2 C K) + C D (4 K - 2) + D  + C ) + B K + A K + D + C
X = ---------------------------------------------------------------------------------------------------------------------------]
                                                              2 K - 2

(C10) /* Cubic polynomial with real and complex roots: */

X^3+3;

                                                             3
(D10)                                                       X  + 3

(C11) SOLVE(%,X);
                                           5/6        1/3        5/6        1/3
                                          3    %I - 3           3    %I + 3              1/3
(D11)                               [X = - ---------------, X = ---------------, X = - 3    ]
                                                  2                    2

(C12) /* A system of 3 simultaneous linear equations in 3 unknowns: */

A*X+B*Y+C*Z = 25;
(D12)                                              C Z + B Y + A X = 25

(C13) 2*X+4*Y+8*Z = 1;
(D13)                                              8 Z + 4 Y + 2 X = 1

(C14) X+Z = 0;
(D14)                                                  Z + X = 0

(C15) SOLVE([%TH(3),%TH(2),%],[X,Y,Z]);
                                     B - 100            C - A - 150                B - 100
(D15)                         [[X = ---------------, Y = ---------------, Z = - ---------------]]
                                    4 C - 6 B - 4 A     4 C - 6 B - 4 A         4 C - 6 B - 4 A

(C16) /*
```

```
A system of 4 simultaneous linear equations in 4 unknowns: */

X-Y+2*Z+W = -5;
(D16)                                          2 Z - Y + X + W = - 5

(C17) -X+3*Z+2*W = 0;
(D17)                                           3 Z - X + 2 W = 0

(C18) 2*X+Y-W = 1;
(D18)                                            Y + 2 X - W = 1

(C19) 2*X+2*Y+Z+3*W = -1;
(D19)                                        Z + 2 Y + 2 X + 3 W = - 1

(C20) SOLVE([%TH(4),%TH(3),%TH(2),%],[X,Y,Z,W]);
                                           64       116       14         53
(D20)                               [[X = - --, Y = ---, Z = --, W = - --]]
                                           41        41       41         41

(C21) /*
```

```
A quartic equation! */

X^4+3*X^2+6*X+10 = 0;
                                                   4     2
(D21)                                             X  + 3 X  + 6 X + 10 = 0

(C22) SOLVE(%,X);
(D22)                                 [X = - %I - 1, X = %I - 1, X = 1 - 2 %I, X = 2 %I + 1]

(C23) /* Non-linear equation using substitution! */

XS:X^2-12*X+3;
                                                        2
(D23)                                                  X  - 12 X + 3

(C24) /* Substitution occurs here! */

SIN(XS)^2-5*SIN(XS)+3;
                                            2  2                       2
(D24)                                    SIN (X  - 12 X + 3) - 5 SIN(X  - 12 X + 3) + 3

(C25) SOLVE(%,X);

Solve is using ARC-trig functions to get a
solution.   Some solutions will be lost.

ATRIG FASL DSK MAXOUT being loaded
Loading done

Solve is using ARC-trig functions to get a
solution.   Some solutions will be lost.
                            SQRT(13) - 5                             SQRT(13) - 5                                 SQRT(13) + 5
(D25) [X = 6 - SQRT(ASIN(- ------------) + 33), X = SQRT(ASIN(- ------------) + 33) + 6, X = 6 - SQRT(ASIN(------------) + 33),
                                 2                                        2                                             2

                                                                                                              SQRT(13) + 5
                                                                                              X = SQRT(ASIN(------------) + 33) + 6]
                                                                                                                    2

(C26) /*
```

```
System of 3 equations of higher degree with 3 unknowns: */

X+Y+Z = 3;
(D26)                                              Z + Y + X = 3

(C27) X*Y+Y*Z+Z*X = -18;
(D27)                                            Y Z + X Z + X Y = - 18

(C28) X^3+Y^3+Z^3 = 189;
                                                   3     3     3
(D28)                                             Z  + Y  + X  = 189

(C29) SOLVE([%TH(3),%TH(2),%] , [X,Y,Z]);

ALGSYS FASL DSK MACSYM being loaded
Loading done

HOMOG FASL DSK MACSYM being loaded
Loading done

MRESUL FASL DSK MAXOUT being loaded
Loading done

SUBRES FASL DSK MAXOUT being loaded
Loading done
(D29) [[X = 0, Y = 6, Z = - 3], [X = 0, Y = - 3, Z = 6], [X = 6, Y = 0, Z = - 3], [X = 6, Y = - 3, Z = 0],
                                                      [X = - 3, Y = 0, Z = 6], [X = - 3, Y = 6, Z = 0]]

(C30) /* Linear system of 3 equations with complex coefficients having 3 unknowns: */

(-5+2*%I)*X1-3*%I*X2+4*X3=7-%I;
(D30)                         4 X3 - 3 %I X2 + (2 %I - 5) X1 = 7 - %I

(C31) (2+%I)*X1+(4-3*%I)*X2+(9-%I)*X3=2;
(D31)                    (9 - %I) X3 + (4 - 3 %I) X2 + (%I + 2) X1 = 2

(C32) 7*%I*X1+5*X2+(1+2*%I)*X3=4+6*%I;
(D32)                      (2 %I + 1) X3 + 5 X2 + 7 %I X1 = 6 %I + 4

(C33) SOLVE([%TH(3),%TH(2),%],[X1,X2,X3]);
                          31462 %I + 20163        126788 %I - 6893          51355 %I + 17451
(D33)              [[X1 = - ----------------, X2 = ----------------, X3 = - ----------------]]
                               67757                   67757                    67757

(C34) /*
```

```
Now, we use nested commands in MACSYMA to convert the complex rational forms of the solutions into their complex
decimal number equivalent forms! */

FIRST(FIRST(%))$

(C35) EXPAND(EV(%,NUMER));
(D35)                                 X1 = - 0.464335788 %I - 0.297578108

(C36) REST(FIRST(%TH(3)),1)$

(C37) FIRST(%)$

(C38) EXPAND(EV(%,NUMER));
(D38)                                  X2 = 1.87121625 %I - 0.101731186

(C39) LAST(%TH(3))$

(C40) EXPAND(EV(%,NUMER));
(D40)                                 X3 = - 0.757929064 %I - 0.257552724

(D41)                                               BATCH DONE

(C42) ^Z
[DDT]
$
```

2.5 Taylor and MacLaurin Series, Limits, Infinite Sums and Products, and Factorials

This section contains a potpourri of MACSYMA's symbolic capabilities. We draw the reader's attention to the interesting fact that one can write subprograms to perform symbolic manipulation in MACSYMA that are somewhat analogous to function subprograms or procedures used in high level programming languages such as FORTRAN or ALGOL. An example is given in this section as the function subprogram MYTAYLOR. However, these subprograms cannot be called or referenced by other programs that are written in FORTRAN or other high level programming languages.

```
(C1) BATCH(DRI10,1,DSK,DRINK)$

(C2) /*

         DEMONSTRATION 2.5 - TAYLOR AND MACLAURIN SERIES, LIMITS, INFINITE SUMS AND PRODUCTS, AND FACTORIALS



TAYLOR AND MACLAURIN SERIES EXPANSION EXAMPLES:

The MACSYMA function command TAYLOR(EXP,VAR,PT,POW) expands the expression EXP in a truncated Taylor series(or Laurent series,
if required) in the variable VAR around the point PT.  The terms through (VAR-PT)^POW are generated.

We begin with a MacLaurin series expansion for SIN(X) expanded to the 13th power term: */

DYNAMALLOC:TRUE$
(D2)                                            TRUE

(C3) TAYLOR(SIN(X),X,0,13)$

HAYAT FASL DSK MACSYM being loaded
Loading done
                                 3    5     7      9        11          13
                                X    X     X      X        X           X
(D3)/T/                     X - -- + --- - ---- + ------ - -------- + ---------- + . . .
                                6    120   5040   362880   39916800   6227020800

(C4) /* Next, we have a MacLaurin series for COS(X) expanded to the 12th power term: */

TAYLOR(COS(X),X,0,12)$
                                  2    4    6      8        10          12
                                 X    X    X      X        X           X
(D4)/T/                      1 - -- + -- - --- + ----- - ------- + --------- + . . .
                                 2    24   720   40320   3628800   479001600

(C5) /* Now we will obtain the MacLaurin series for TAN(X) expanded to the 13th power term: */

TAYLOR(TAN(X),X,0,13)$

EULBRN FASL DSK MAXOUT being loaded
Loading done
                                 3      5       7      9         11          13
                                X    2 X    17 X    62 X    1382 X     21844 X
(D5)/T/                     X + -- + ---- + ----- + ----- + --------- + ---------- + . . .
                                3     15     315    2835    155925     6081075

(C6) /* Using the fundamental identity, SIN(X)/COS(X)=TAN(X), we can also obtain the MacLaurin series for TAN(X) by dividing
the SIN(X) series expansion by the COS(X) series expansion given above: */

%TH(3)/%TH(2)$
                                 3      5       7      9         11          13
                                X    2 X    17 X    62 X    1382 X     21844 X
(D6)/T/                     X + -- + ---- + ----- + ----- + --------- + ---------- + . . .
                                3     15     315    2835    155925     6081075

(C7) /*
```

```
Next, we will compute the term by term product series of the MacLaurin SIN(X) and COS(X) series: */

%TH(4)*%TH(3);
```

(D7)/T/

$$X - \frac{2X^3}{3} + \frac{2X^5}{15} - \frac{4X^7}{315} + \frac{2X^9}{2835} - \frac{4X^{11}}{155925} + \frac{4X^{13}}{6081075} + \ldots$$

```
(C8) /* Now we will differentiate this product series, term by term, with respect to X to yield the MacLaurin series
for COS(X)^2-SIN(X)^2 which is the derivative of SIN(X)*COS(X) with respect to X: */

DIFF(%,X);
```

(D8)/T/

$$1 - 2X^2 + \frac{2X^4}{3} - \frac{4X^6}{45} + \frac{2X^8}{315} - \frac{4X^{10}}{14175} + \frac{4X^{12}}{467775} + \ldots$$

```
(C9) /* Next, we integrate this MacLaurin series expansion, term by term, with respect to X to obtain the first 7 terms of the
MacLaurin series expansion for SIN(X)*COS(X) in D7.  */

INTEGRATE(%,X);

SIN FASL DSK MACSYM being loaded
Loading done

SININT FASL DSK MACSYM being loaded
Loading done

SCHATC FASL DSK MACSYM being loaded
Loading done
```

(D9)

$$\frac{4X^{13}}{6081075} - \frac{4X^{11}}{155925} + \frac{2X^9}{2835} - \frac{4X^7}{315} + \frac{2X^5}{15} - \frac{2X^3}{3} + X$$

```
(C10) /* For this next expression, we will attempt to obtain a MacLaurin series expansion to the 12th power term for: */

1/(COS(X)-SEC(X))^3;
```

(D10)

$$\frac{1}{(COS(X) - SEC(X))^3}$$

```
(C11) /* However, we actually obtain a finite Laurent series expansion to the 12th positive power term: */

TAYLOR(%,X,0,12);
```

(D11)/T/

$$- \frac{1}{X^6} + \frac{1}{2X^4} + \frac{11}{120X^2} - \frac{347}{15120} - \frac{6767X^2}{604800} - \frac{15377X^4}{7983360} - \frac{977297X^6}{31135104000} + \frac{682163X^8}{7925299200} + \frac{16091678051X^{10}}{485028311040000} + \frac{241778832797X^{12}}{28071946248192000} + \ldots$$

```
(C12) /*
```

```
In this next example, we are going to obtain a Taylor series expansion in X for the expression: */
SIN(X)/X;
                                                SIN(X)
(D12)                                           ------
                                                  X
(C13) /* This Taylor series will be expanded about the point X=A to the 7th power term: */
TAYLOR(%,X,A,7);
                                                                  2                                 2
          SIN(A)   (COS(A) A - SIN(A)) (X - A)   (SIN(A) A  + 2 COS(A) A - 2 SIN(A)) (X - A)
(D13)/T/ ------ + --------------------------- - -------------------------------------------
           A                  2                                      3
                             A                                    2 A

               3            2                                3
   (COS(A) A  - 3 SIN(A) A  - 6 COS(A) A + 6 SIN(A)) (X - A)
 - ---------------------------------------------------------
                                4
                             6 A

               4            3             2                                 4
   (SIN(A) A  + 4 COS(A) A  - 12 SIN(A) A  - 24 COS(A) A + 24 SIN(A)) (X - A)
 + --------------------------------------------------------------------------
                                       5
                                   24 A

               5            4             3             2                                   5
   (COS(A) A  - 5 SIN(A) A  - 20 COS(A) A  + 60 SIN(A) A  + 120 COS(A) A - 120 SIN(A)) (X - A)
 + -------------------------------------------------------------------------------------------
                                            6
                                       120 A

               6            5             4              3              2                                   6
   (SIN(A) A  + 6 COS(A) A  - 30 SIN(A) A  - 120 COS(A) A  + 360 SIN(A) A  + 720 COS(A) A - 720 SIN(A)) (X - A)
 - ----------------------------------------------------------------------------------------------------------
                                                  7
                                             720 A

              7            6             5              4              3               2
 - (COS(A) A  - 7 SIN(A) A  - 42 COS(A) A  + 210 SIN(A) A  + 840 COS(A) A  - 2520 SIN(A) A  - 5040 COS(A) A + 5040 SIN(A))

        7        8
 (X - A) /(5040 A ) + . . .
(C14) /*
```

```
Then, we will evaluate this series expansion for A=%PI/2: */

EV(%,A = %PI/2);
                 6          4            2              7             7            5              3               6
(D14)/R/ ((64 %PI  - 7680 %PI  + 368640 %PI  - 2949120) X  + (- 256 %PI  + 30720 %PI  - 1474560 %PI  + 11796480 %PI) X

           8           6             4               2   5          9            7              5               3   4
 + (432 %PI  - 53760 %PI  + 2580480 %PI  - 20643840 %PI ) X  + (- 400 %PI  + 53760 %PI  - 2580480 %PI  + 20643840 %PI ) X

           10          8             6               4   3         11           9             7              5   2
 + (220 %PI   - 33120 %PI  + 1612800 %PI  - 12902400 %PI ) X  + (- 72 %PI   + 12480 %PI  - 645120 %PI  + 5160960 %PI ) X

          12          10            8              6            13          11            9              7              8
 + (13 %PI   - 2640 %PI   + 155520 %PI  - 1290240 %PI ) X - %PI   + 240 %PI   - 17280 %PI  + 184320 %PI )/(11520 %PI )

(C15) /* Using this Taylor series expansion, let's compute SIN(X)/X for X=%PI/2: */

EV(%,X = %PI/2);
                                                            2
(D15)/R/                                                   ---
                                                           %PI

(C16) /* We can convert this result to a decimal number: */

EV(%,NUMER);
(D16)                                                  0.63661978

(C17) /* For verification of the above result, we directly evaluate the expression: */

SIN(X)/X;
                                                         SIN(X)
(D17)                                                    ------
                                                           X

(C18) /* for X=%PI/2: */

EV(%,X = %PI/2);
                                                            2
(D18)                                                      ---
                                                           %PI

(C19) /* Here is an example of a user-defined Taylor series expansion function which expands the expression EXPR in a truncated
Taylor series in the variable VAR around the point POINT.  The terms through (VAR-POINT)^HIPOWER are generated.  The
function routine generates the series:
                                   F(X)=F(A)+F'(A)(X-A)+(F''(A)(X-A)^2)/2!+(F'''(A)(X-A)^3)/3!+...
in an iterative manner.  For details concerning the actual coding of the routine, see MACSYMA Reference Manual[1].
*/

MYTAYLOR(EXPR,VAR,POINT,HIPOWER):=BLOCK([RESULT],RESULT:SUBST(POINT,VAR,EXPR),
FOR I THRU HIPOWER DO [EXPR:DIFF(EXPR,VAR)/I,RESULT:RESULT+(VAR-POINT)^I*SUBST(POINT,VAR,EXPR)],RETURN(RESULT));
(D19) MYTAYLOR(EXPR, VAR, POINT, HIPOWER) := BLOCK([RESULT], RESULT : SUBST(POINT, VAR, EXPR),

                                  DIFF(EXPR, VAR)                                  I
         FOR I THRU HIPOWER DO [EXPR : ---------------, RESULT : RESULT + (VAR - POINT)  SUBST(POINT, VAR, EXPR)], RETURN(RESULT))
                                         I

(C20) /*
```

This next expression is called a generating function for Chebeshev polynomials of the first kind! */

(1-U*X)/(1-2*U*X+U^2)$

(D20)
$$\frac{1 - U X}{- 2 U X + U^{2} + 1}$$

(C21) /* Using our own defined Taylor series expansion function above, we will generate the Chebeshev polynomials as coefficients to the U terms in the series expansion of the generating function to the 10th power term: */

MYTAYLOR(MYTAYLOR(%,X,0,10),U,0,10)$

(D21)
$$\frac{U^{10} (1857945600 X^{10} - 4644864000 X^{8} + 4064256000 X^{6} - 1451520000 X^{4} + 181440000 X^{2} - 3628800)}{3628800}$$
$$+ \frac{U^{9} (92897280 X^{9} - 209018880 X^{7} + 156764160 X^{5} - 43545600 X^{3} + 3265920 X)}{362880}$$
$$+ \frac{U^{8} (5160960 X^{8} - 10321920 X^{6} + 6451200 X^{4} - 1290240 X^{2} + 40320)}{40320} + \frac{U^{7} (322560 X^{7} - 564480 X^{5} + 282240 X^{3} - 35280 X)}{5040}$$
$$+ \frac{U^{6} (23040 X^{6} - 34560 X^{4} + 12960 X^{2} - 720)}{720} + \frac{U^{5} (1920 X^{5} - 2400 X^{3} + 600 X)}{120} + \frac{U^{4} (192 X^{4} - 192 X^{2} + 24)}{24} + \frac{U^{3} (24 X^{3} - 18 X)}{6}$$
$$+ \frac{U^{2} (4 X^{2} - 2)}{2} + U X + 1$$

(C22) /* We can now use RATSIMP to simplify the Chebeshev polynomial coefficients. */

RATSIMP(%,U)$

(D22)
$$U^{10} (512 X^{10} - 1280 X^{8} + 1120 X^{6} - 400 X^{4} + 50 X^{2} - 1) + U^{9} (256 X^{9} - 576 X^{7} + 432 X^{5} - 120 X^{3} + 9 X)$$
$$+ U^{8} (128 X^{8} - 256 X^{6} + 160 X^{4} - 32 X^{2} + 1) + U^{7} (64 X^{7} - 112 X^{5} + 56 X^{3} - 7 X) + U^{6} (32 X^{6} - 48 X^{4} + 18 X^{2} - 1)$$
$$+ U^{5} (16 X^{5} - 20 X^{3} + 5 X) + U^{4} (8 X^{4} - 8 X^{2} + 1) + U^{3} (4 X^{3} - 3 X) + U^{2} (2 X^{2} - 1) + U X + 1$$

(C23) /*

```
This next expression provides another example of a multivariate Taylor series expansion: */

SIN(X*Y)+SIN(X);
(D23)                                       SIN(X Y) + SIN(X)

(C24) /* There are several ways of obtaining Taylor series expansions for multivariate functions.  Here we demonstrate one such
method in which we expand the X terms about the point %PI/2 and the Y terms about the point 0: */

TAYLOR(%,[X,%PI/2,6],[Y,0,6]);
                        3  3       5  5             2  3      4  5                                             3       3  5
                %PI Y  %PI  Y   %PI  Y            %PI  Y   %PI  Y                    %PI        1   %PI Y   %PI  Y                  %PI 2
(D24)/T/ 1 + ----- - ------- + ------- + (Y - ------- + ------- + . . .) (X - ---) + (- - - ------ + ------- + . . .) (X - ---)
               2       48      3840              8        384                     2         2     4        96                      2

                                                                                                          5     %PI 5        %PI 6
                         3       2  5                                          5                          Y  (X - ---)    (X - ---)
                         Y    %PI  Y                      %PI 3    1     %PI Y                    %PI 4              2            2
                  + (- -- + ------- + . . .) (X - ---) + (-- + ------ + . . .) (X - ---)  + ------------- - ---------- + . . .
                         6      48                          2      24     48                     2            120           720

(C25) /*
```

```
LIMIT EXAMPLES:

The MACSYMA function command LIMIT(EXP,VAR,VAL,DIR) finds the limit of EXP as the real variable VAR approaches the value VAL
from the direction DIR.  DIR may have the value PLUS for a limit from above, MINUS for a limit from below, or may be omitted
(implying a two-sided limit is to be computed).  LIMIT also uses the special symbols: INF(positive infinity) and
MINF(negative infinity).  In the case of indeterminant forms, L'Hospital's rule is used up to a maximum of 4 times.  This
prevents infinite looping which could occur in certain cases.  Let's begin by defining the following function: */

I(R):=(E*(1-%E^(-(R*T)/L)))/R;
                                                                  R T
                                                                - ---
                                                                   L
                                                        E (1 - %E     )
(D25)                                           I(R) := ---------------
                                                               R

(C26) /* Now, compute the limit of this function as R approaches 0: */

'LIMIT(I(R),R,0);
                                                                  R T
                                                                - ---
                                                                   L
                                                          1 - %E
(D26)                                            E (limit -----------)
                                                    R -> 0      R

(C27) %,LIMIT;

LIMIT FASL DSK MACSYM being loaded
Loading done

RPART FASL DSK MACSYM being loaded
Loading done
                                                           E T
(D27)                                                      ---
                                                            L

(C28) 'LIMIT(SIN(X)/X,X,0);
                                                            SIN(X)
(D28)                                                 limit ------
                                                      X -> 0   X

(C29) %,LIMIT;

TRIGEX FASL DSK MAXOUT being loaded
Loading done
(D29)                                                       1

(C30) 'LIMIT(LOG(SIN(X))/LOG(TAN(X)),X,0,PLUS);
                                                          LOG(SIN(X))
(D30)                                               limit  -----------
                                                    X -> 0+ LOG(TAN(X))

(C31) %,LIMIT;
(D31)                                                       1

(C32) /*
```

```
*/

'LIMIT((2*X-1)/(3*X+5),X,INF);

                                                      2 X - 1
(D32)                                        limit    -------
                                             X -> INF 3 X + 5

(C33) %,LIMIT;

                                                         2
(D33)                                                    -
                                                         3

(C34) 'LIMIT(TAN(X)^COS(X),X,%PI/2,MINUS);
                                                                  COS(X)
(D34)                                      limit      TAN(X)
                                                 %PI
                                           X -> ----
                                                  2

(C35) %,LIMIT;
(D35)                                                    1

(C36) 'LIMIT((1-COS(X))/X^2,X,0);
                                                      1 - COS(X)
(D36)                                         limit   ----------
                                              X -> 0       2
                                                          X

(C37) %,LIMIT;
                                                         1
(D37)                                                    -
                                                         2

(C38) 'LIMIT(X^SIN(X),X,0,PLUS);
                                                            SIN(X)
(D38)                                         limit    X
                                              X -> 0+

(C39) %,LIMIT;
(D39)                                                    1

(C40) 'LIMIT(SIN(X)^2/(1-SEC(X)),X,0);
                                                            2
                                                         SIN (X)
(D40)                                         limit   ----------
                                              X -> 0  1 - SEC(X)

(C41) %,LIMIT;
(D41)                                                   - 2

(C42) /*
```

```
INFINITE AND FINITE SUMS AND PRODUCTS EXAMPLES:

The MACSYMA function command SUM(EXP,IND,LO,HI) performs a summation of the values of EXP as the index IND varies from LO to HI.
If the upper and lower limits differ by an integer, then each term in the sum is evaluated and added.  If the upper and
lower limits differ by a variable, then the sum is simplified provided SIMPSUM[FALSE] is TRUE.  We define a function G[N](X): */

G[N](X):='SUM(I*X^I,I,0,N);
                                                     N
                                                    ====
                                                    \         I
(D42)                                      G (X) :=  >     I X
                                            N       /
                                                    ====
                                                    I = 0

(C43) /* Its infinite sum is */

G[INF](X);
                                                  INF
                                                  ====
                                                  \         I
(D43)                                              >     I X
                                                  /
                                                  ====
                                                  I = 0

(C44) /* The finite sum of its 1st 10 terms is */

G[10](X);
                                                   10
                                                  ====
                                                  \         I
(D44)                                              >     I X
                                                  /
                                                  ====
                                                  I = 0

(C45) /* Now let's evaluate this sum: */

EV(%,SUM);
                                   10       9       8       7       6       5       4       3       2
(D45)                          10 X   + 9 X  + 8 X  + 7 X  + 6 X  + 5 X  + 4 X  + 3 X  + 2 X  + X

(C46) /* Here we display the sum of the 1st 4 values of J! in conventional mathematical notation: */

'SUM(J!,J,1,4);
                                                   4
                                                  ====
                                                  \
(D46)                                              >     J!
                                                  /
                                                  ====
                                                  J = 1

(C47) /*
```

```
This is an example where the upper and lower limits differ by an integer.  We evaluate the sum: */

SUM(J!,J,1,4);
(D47)                                                  33

(C48) /* Verification of the sum: */

4!+3!+2!+1!;
(D48)                                                  33

(C49) /* Now, let's see the effect of setting SIMPSUM[FALSE]:TRUE.  In each case we will first display the sum with SIMPSUM
off and then enter the same sum command after the SIMPSUM switch is turned on. */

SUM(I^2+2^I,I,0,N);
                                               N
                                              ====
                                              \        I    2
(D49)                                          >     (2  + I )
                                              /
                                              ====
                                              I = 0

(C50) SIMPSUM:TRUE;
(D50)                                                 TRUE

(C51) SUM(I^2+2^I,I,0,N);

SUM FASL DSK MAXOUT being loaded
Loading done

BINOML FASL DSK MAXOUT being loaded
Loading done
                                                    3      2
                                           N + 1   2 N  + 3 N  + N
(D51)                                     2      + --------------- - 1
                                                          6

(C52) SIMPSUM:FALSE;
(D52)                                                FALSE

(C53) SUM(3^(-I),I,1,INF);
                                                INF
                                                ====
                                                \     1
(D53)                                            >    --
                                                /      I
                                                ====  3
                                                I = 1

(C54) SIMPSUM:TRUE;
(D54)                                                 TRUE

(C55) SUM(3^(-I),I,1,INF);
                                                       1
(D55)                                                  -
                                                       2

(C56) /*
```

```
*/

SIMPSUM:FALSE;
(D56)                                                      FALSE

(C57) SUM(I^2,I,1,4)*SUM(1/I^2,I,1,INF);
                                                           INF
                                                           ====
                                                           \      1
(D57)                                                   30  >    --
                                                           /      2
                                                           ====   I
                                                           I = 1

(C58) SIMPSUM:TRUE;
(D58)                                                       TRUE

(C59) SUM(I^2,I,1,4)*SUM(1/I^2,I,1,INF);

ZETA FASL DSK MAXOUT being loaded
Loading done
                                                                2
(D59)                                                      5 %PI

(C60) SIMPSUM:FALSE;
(D60)                                                      FALSE

(C61) SUM(I^4,I,1,N);
                                                          N
                                                          ====
                                                          \      4
(D61)                                                      >    I
                                                          /
                                                          ====
                                                          I = 1

(C62) SIMPSUM:TRUE;
(D62)                                                       TRUE

(C63) SUM(I^4,I,1,N);
                                                       5       4        3
                                                    6 N  + 15 N  + 10 N  - N
(D63)                                               ------------------------
                                                              30

(C64) FACTOR(%);
                                                                        2
                                                N (N + 1) (2 N + 1) (3 N  + 3 N - 1)
(D64)                                           ------------------------------------
                                                                 30

(C65) SIMPSUM:FALSE;
(D65)                                                      FALSE

(C66) /*
```

```
Euler's constant will be used in the next example! */

%EULER=0.5772156649;
(D66)                                           %EULER = 0.577215664

(C67) /* The MACSYMA function command PRODUCT(EXP,IND,LO,HI) gives the product of the values of EXP as the index IND varies from
LO to HI.  The evaluation is similar to that of SUM.  The Gamma function can be represented as an infinite product as follows: */

1/GAMMA(X)=X*%E^(%EULER*X)*'PRODUCT(((1+X/M)*%E^(-X/M)),M,1,INF);

GAMMA FASL DSK MAXOUT being loaded
Loading done

PRODCT FASL DSK MAXOUT being loaded
Loading done
                                                                        INF
                                                                       /===\
                                             1             %EULER X     ! !     X         - X/M
(D67)                                     -------- = X %E                ! !    (- + 1) %E
                                          GAMMA(X)                       ! !     M
                                                                        M = 1

(C68) /* %PI/2 can be represented as an infinite product called the Wallis product! */

%PI/2='PRODUCT(((2*I)/(2*I-1))*((2*I)/(2*I+1)),I,1,INF);
                                                      INF
                                                     /===\           2
                                                %PI   ! !         4 I
(D68)                                           --- = ! !  -------------------
                                                 2    ! !  (2 I - 1) (2 I + 1)
                                                     I = 1

(C69) /* Let's approximate %PI/2 using the Wallis product as a finite product of 500 terms: */

%PI/2='PRODUCT(((2*I)/(2*I-1))*((2*I)/(2*I+1)),I,1,500);
                                                      500
                                                     /===\           2
                                                %PI   ! !         4 I
(D69)                                           --- = ! !  -------------------
                                                 2    ! !  (2 I - 1) (2 I + 1)
                                                     I = 1

(C70) /* We evaluate this product numerically and obtain accuracy to 3 decimal places! */

%PI/2=EV(EV(PART(%,2),PRODUCT),NUMER);
                                                      %PI
(D70)                                                 --- = 1.57001175
                                                       2

(C71) /*
```

```
FACTORIAL EXAMPLES:

Here is a user recursive definition for N!: */

FAC(N):=IF N=0 THEN 1 ELSE N*FAC(N-1);
(D71)                        FAC(N) := IF N = 0 THEN 1 ELSE N FAC(N - 1)

(C72) /* In each of the following examples, we compute a factorial using both the user-defined function FAC(N) and the
factorial operator '!': */

FAC(5);
(D72)                                              120

(C73) 5!;
(D73)                                              120

(C74) FAC(8);
(D74)                                             40320

(C75) 8!;
(D75)                                             40320

(C76) FAC(20);
(D76)                                       2432902008176640000

(C77) 20!;
(D77)                                       2432902008176640000

(C78) FAC(50);
(D78)                30414093201713378043612608166064768844377641568960512000000000000

(C79) 50!;
(D79)                30414093201713378043612608166064768844377641568960512000000000000

(D80)                                           BATCH DONE

(C81) ^Z
[DDT]
*
```

3. Numerical Methods

Since MACSYMA's basic purpose is symbolic manipulation, the inclusion of numerical facilities was not intended to provide routines that are callable from scientific application programs developed in a high order programming language like FORTRAN on computers such as the UNIVAC 1108 or VAX-11/780. However, occasion may arise where a computational problem can be solved conveniently within the MACSYMA environment using MACSYMA's numerical library. Some of MACSYMA's numerical techniques are

1) Numerical integration using either Romberg's or Simpson's methods;

2) Fast Fourier transforms and their inverses, which also include polar-to-rectangular and rectangular-to-polar transformation functions for converting magnitude and phase arrays into real/imaginary form;

3) Roots of equations by interpolation;

4) Bessel functions in their various forms and orders;

5) Runge-Kutta method for solving differential equations; and

6) Polynomial zeros (isolation of zeros of a polynomial based on the Collins-Loos differentiation algorithm).

3.1 Illustrations of Numerical Computations Using MACSYMA

The first section of this final demonstration deals with numerical integration using the Romberg function in MACSYMA. The second section of examples deals with the numerical solution of algebraic equations using our own user-defined Newton-Raphson algorithm. Finally, MACSYMA's capability for large number representation is demonstrated.

In the Romberg integration demonstrations, TRANSLATE is a *preprocessor that saves the user the effort of learning the* LISP language in order to code the integrand function definition into internal LISP expressions. This results in a gain in execution speed when the function is later referenced. However, the user will have to learn the MACSYMA language to define integrands whose definitions depend on the subinterval in which the value of the variable of integration falls. Otherwise, TRANSLATE cannot process them properly.

```
(C1) BATCH(DRI12,1,DSK,DRINK);

(C2) /*

                              DEMONSTRATION 3.1 - NUMERICAL METHODS


NUMERICAL INTEGRATION EXAMPLES:


The function ROMBERG(FN,LL,UL) or ROMBERG(EXP,VAR,LL,UL) will perform numerical integration in MACSYMA.  The arguments to ROMBERG
are described as ROMBERG(<function name>,<lower limit>,<upper limit>) or ROMBERG(<integrand expression>,<variable of
integration>,<lower limit>,<upper limit>), respectively.  Although it is not absolutely necessary, translating or
compiling the integrand function definitions gives more efficiency.  This is accomplished through the use of:
                          1)MODEDECLARE(Y1,MODE1,Y2,MODE2,...) which declares the modes of variables and functions
                            for subsequent translation or compilation of functions, and
                          2)TRANSLATE(F1,F2,...) which translates the user defined functions F1,F2,..., from the MACSYMA
                            language into their internal LISP language representation.
This results in a gain in speed when the function is called.

The accuracy of the integration is governed by the global variables ROMBERGTOL[default value 1.0E-4] and ROMBERGIT[default value
11].  ROMBERG will return a result if the relative difference in successive approximations is less than ROMBERGTOL.  It will try
halving the stepsize ROMBERGIT times before it gives up.  The user may re-define ROMBERTOL and ROMBERGIT if greater or less
accuracy is desired than normally provided by the default values.

ROMBERG may be called recursively and thus can do double and triple iterated integrals.  (It is all the more important to
TRANSLATE the functions in this case.)


EXAMPLES:

Since many MACSYMA library routines must be loaded into core in order to evaluate definite integrals symbolically,
the examples used in this demonstration have been computed symbolically and their values computed numerically in a separate run.
To save core space for this present demonstration only the results are displayed. */

DYNAMALLOC:TRUE;
(D2)                                                          TRUE

(C3) /* Display of example (1) and symbolically computed result: */

'INTEGRATE(1/X,X,1,137.2) = 4.921439777;
                                                    137.2
                                                   /
                                                   [       1
(D3)                                               I       - dX = 4.92143977
                                                   ]       X
                                                   /
                                                    1

(C4) /* This defines the function F(X)=1/X in floating point mode. */

F(X):=(MODEDECLARE([FUNCTION(F),X],FLOAT),1/X);
                                                                          1
(D4)                                     F(X) := (MODE_DECLARE([FUNCTION(F), X], FLOAT), -)
                                                                          X

(C5) /*
```

```
Translate function F: */

TRANSLATE(F);

TRANSS FASL DSK MACSYM being loaded
TRANSL AUTOLO DSK MACSYM being loaded
Loading done
Loading done

TRANSL FASL DSK MACSYM being loaded
MLOAD FASL DSK MACSYM being loaded
Loading done
TRDATA FASL DSK MACSYM being loaded
TRMODE FASL DSK MACSYM being loaded
Loading done
Loading done
Loading done

TROPER FASL DSK MACSYM being loaded
Loading done

TRUTIL FASL DSK MACSYM being loaded
Loading done

MDEFUN FASL DSK MACSYM being loaded
Loading done
(D5)                                                            DONE

(C6) /* Set ROMBERGTOL to 5.0e-6: */

ROMBERGTOL:5.0E-6;
(D6)                                                            5.0E-6

(C7) /* Lower limit of integration is */

LL:1;
(D7)                                                            1

(C8) /* Upper limit of integration is */

UL:137.2;
(D8)                                                            137.2

(C9) /* Evaluate example (1) using Romberg integration: */

ROMBERG(F,LL,UL);

ROMBRG FASL DSK MACSYM being loaded
Loading done

NUMAPL FASL DSK MACSYM being loaded
Loading done
(D9)                                                            4.9214393

(C10) /*
```

```
Display of example (2) and symbolically computed results: */

'INTEGRATE(X^2*%E^X,X,0,2) = 2*(%E^2-1);
                                                 2
                                                /
                                                [   2   X            2
(D10)                                           I  X  %E  dX = 2 (%E   - 1)
                                                ]
                                                /
                                                 0

(C11) /* Re-display with the numerical decimal conversion of the result for comparison with the Romberg integration: */

PART(%,1) = EV(PART(%,2),NUMER);
                                                 2
                                                /
                                                [   2   X
(D11)                                           I  X  %E  dX = 12.7781123
                                                ]
                                                /
                                                 0

(C12) /* Define G(X)=X^2*%E^X in floating point mode: */

G(X):=(MODEDECLARE([FUNCTION(G),X],FLOAT),X^2*%E^X);
                                                                       2   X
(D12)                       G(X) := (MODE_DECLARE([FUNCTION(G), X], FLOAT), X  %E )

(C13) /* Translate the function G: */

TRANSLATE(G);
(D13)                                                   DONE

(C14) /* Set lower limit of integration: */

LL:0;
(D14)                                                    0

(C15) /* Set upper limit of integration: */

UL:2;
(D15)                                                    2

(C16) /* Evaluate the integral by the Romberg method: */

ROMBERG(G,LL,UL);

FCALL FASL DSK MACSYM being loaded
Loading done
(D16)                                                 12.7781124

(C17) /*
```

```
Display of example (3) and symbolically computed result: */

'INTEGRATE(1/(1+X^2),X,0,1) = %PI/4;
                                                          1
                                                         /
                                                         [      1              %PI
(D17)                                                    I   ------ dX    =    ---
                                                         ]    2                 4
                                                         /   X  + 1
                                                          0

(C18) /* Convert result to decimal number: */

PART(%,1) = EV(PART(%,2),NUMER);
                                                      1
                                                     /
                                                     [      1
(D18)                                                I   ------ dX = 0.78539816
                                                     ]    2
                                                     /   X  + 1
                                                      0

(C19) /* Define H(X)=1/(1+X^2) in floating point mode: */

H(X):=(MODEDECLARE([FUNCTION(H),X],FLOAT),1/(1+X^2));
                                                                                   1
(D19)                     H(X) := (MODE_DECLARE([FUNCTION(H), X], FLOAT), ------)
                                                                                    2
                                                                             1 + X

(C20) /* Translate the function H: */

TRANSLATE(H);
(D20)                                                          DONE

(C21) /* Set lower limit of integration: */

LL:0;
(D21)                                                          0

(C22) /* Set upper limit of integration: */

UL:1;
(D22)                                                          1

(C23) /* Evaluate the integral by the Romberg method: */

ROMBERG(H,LL,UL);
(D23)                                                     0.785398155

(C24) /*
```

```
...ion of example (4) and symbolically computed result which in this case is a double iterated integral: */

INTEGRATE('INTEGRATE(X*Y/(X+Y),Y,0,X/2),X,1,3) = 13/3*(2*LOG(2/3)+1);
                                       X
                                       -
                                  3    2                       2
                                  /    /                 13 (2 LOG(-) + 1)
                                  [    [    Y                      3
(D24)                             I  X I  ----- dY dX = -----------------
                                  ]    ]  Y + X                 3
                                  /    /
                                  1    0

(C25) /* Convert the result to a decimal number: */

PART(%,1) = EV(PART(%,2),NUMER);
                                         X
                                         -
                                    3    2
                                    /    /
                                    [    [    Y
(D25)                               I  X I  ----- dY dX = 0.819302335
                                    ]    ]  Y + X
                                    /    /
                                    1    0

(C26) /* Note the double function and mode definitions and the particular way the total integrand is defined in the next two
steps: */

R(Y):=(MODEDECLARE([FUNCTION(R),X,Y],FLOAT),X*Y/(X+Y));
                                                                                 X Y
(D26)                    R(Y) := (MODE_DECLARE([FUNCTION(R), X, Y], FLOAT), -----)
                                                                               X + Y

(C27) S(X):=(MODEDECLARE([FUNCTION(R,S),X],FLOAT),ROMBERG(R,0,X/2));
                                                                                            X
(D27)                 S(X) := (MODE_DECLARE([FUNCTION(R, S), X], FLOAT), ROMBERG(R, 0, -))
                                                                                            2

(C28) /* Now, we translate both functions R and S: */

TRANSLATE(R,S);


(D28)                                           DONE

(C29) /* Note that we perform the Romberg integration on the function S recursively and that the lower and upper limits of
integration are entered as integer arguments as a matter of convenience. */

ROMBERG(S,1,3);
(D29)                                         0.8193024

(C30) KILL(LABELS,VALUES);
(D0)                                            DONE

(C1) /*
```

```
NUMERICAL SOLUTION OF ALGEBRAIC EQUATIONS:

We will solve five equations using the Newton-Raphson algorithm.  We have defined our own Newton-Raphson algorithm using the
MACSYMA language.  The user-defined function NEWTON(EXP,VAR,X0,EPS) will solve the equation: EXP=0 for the variable VAR using X0
as an initial solution estimate.  EPS is the error criterion for determining the number of significant decimal places the
user desires.  A typical value for EPS might be 5.0E-6(OR 5.0[10^-6]).  The Newton-Raphson iterative formula is expressed as
follows:
                                                                      F(X )
                                                                         i
                                                  X         =   X   -   -------
                                                   i+1           i       F'(X )
                                                                             i

Again, as in the MYTAYLOR user-defined routine, the code is essentially performing this iterative process and we will omit
further detail. */

NEWTON(EXP,VAR,X0,EPS):=BLOCK([XN,S,NUMER],NUMER:TRUE,S:DIFF(EXP,VAR),XN:X0,LOOP,XNP1:XN-SUBST(XN,VAR,EXP)/SUBST(XN,VAR,S),
IF ABS(XNP1-XN)+ABS(SUBST(XNP1,VAR,EXP)) < EPS THEN RETURN(XNP1) ELSE XN:XNP1,GO(LOOP));
(D1) NEWTON(EXP, VAR, X0, EPS) := BLOCK([XN, S, NUMER], NUMER : TRUE, S : DIFF(EXP, VAR), XN : X0, LOOP,

              SUBST(XN, VAR, EXP)
 XNP1 : XN - -------------------, IF ABS(XNP1 - XN) + ABS(SUBST(XNP1, VAR, EXP)) < EPS THEN RETURN(XNP1) ELSE XN : XNP1, GO(LOOP))
               SUBST(XN, VAR, S)

(C2) /* The 1st equation is */

SQRT(Y+7)-SQRT(2*Y-9) = 1;
(D2)                                        SQRT(Y + 7) - SQRT(2 Y - 9) = 1

(C3) /* Initial solution estimate is */

Y0:7.0;
(D3)                                                     7.0

(C4) /* Error criterion is */

EPS:5.0E-6;
(D4)                                                    5.0E-6

(C5) /* Write the equation in the form where it is equal to zero! */

EQN:PART(%TH(3),1)-PART(%TH(3),2);
(D5)                                       - SQRT(2 Y - 9) + SQRT(Y + 7) - 1

(C6) /* Solve the equation for variable Y by the Newton-Raphson method: */

NEWTON(EQN,Y,Y0,EPS);
(D6)                                                   9.0000002

(C7) /*
```

```
Re-solve by the Newton-Raphson method entering as a new root estimate one based on the previous calculation and this
time we obtain the exact solution numerically. */

YO:9.0;
(D7)                                                        9.0

(C8) NEWTON(EQN,Y,YO,EPS);
(D8)                                                        9.0

(C9) /* The 2nd equation is */

1/V-2/V^2+8/V^3+192/V^4 = 1;
                                                   1   2    8    192
(D9)                                               - - -- + -- + --- = 1
                                                   V    2    3     4
                                                       V    V     V

(C10) /* Re-write in the proper form: */

EQN:PART(%,1)-PART(%,2);
                                                   1   2    8    192
(D10)                                              - - -- + -- + --- - 1
                                                   V    2    3     4
                                                       V    V     V

(C11) /* Initial solution estimate is */

VO:3.0;
(D11)                                                       3.0

(C12) /* Solve numerically by the Newton-Raphson method: */

NEWTON(EQN,V,VO,EPS);
(D12)                                                       4.0

(C13) /* The 3rd equation is */

EQN:R-1/2-SIN(R)=0;
                                                                1
(D13)                                              - SIN(R) + R - - = 0
                                                                2

(C14) EQN:PART(%,1);
                                                                 1
(D14)                                               - SIN(R) + R - -
                                                                 2

(C15) RO:1.45;
(D15)                                                       1.45

(C16) NEWTON(EQN,R,RO,EPS);
(D16)                                                     1.4973004

(C17) RO:1.49;
(D17)                                                       1.49

(C18) NEWTON(EQN,R,RO,EPS);
(D18)                                                     1.49730039

(C19) /*
```

```
The 4th equation is */

EQN:2^S-4*S=0;
                                          S
(D19)                                    2  - 4 S = 0

(C20) EQN:PART(%,1);
                                          S
(D20)                                    2  - 4 S

(C21) S0:0.2;
(D21)                                      0.2

(C22) NEWTON(EQN,S,S0,EPS);
(D22)                                    0.309906933

(C23) S0:0.3;
(D23)                                      0.3

(C24) NEWTON(EQN,S,S0,EPS);
(D24)                                    0.309906933

(C25) /* Another root estimate (a 2nd root): */

S0:3.6;
(D25)                                      3.6

(C26) NEWTON(EQN,S,S0,EPS);
(D26)                                      4.0

(C27) /* The 5th equation is */

EQN:%E^T-4*T=0;
                                          T
(D27)                                    %E  - 4 T = 0

(C28) EQN:PART(%,1);
                                          T
(D28)                                    %E  - 4 T

(C29) T0:0.35;
(D29)                                      0.35

(C30) NEWTON(EQN,T,T0,EPS);
(D30)                                    0.35740295

(C31) T0:2.15;
(D31)                                      2.15

(C32) NEWTON(EQN,T,T0,EPS);
(D32)                                    2.15329236

(C33) /*
```

LARGE NUMBER REPRESENTATION:

MACSYMA has a feature which allows for virtually unlimited (up to one's list space capacity) floating point precision. BFLOAT(X) converts all numbers and functions of numbers to bigfloat numbers. Setting FPPREC[16] to N overides the default bigfloat precision re-setting it to N digits.

```
EXAMPLES: */

FPPREC:50;

FLOAT FASL DSK MACSYM being loaded
Loading done
(D33)                                               50

(C34) BFLOAT(%PI);
(D34)                        3.1415926535897932384626433832795028841971693993751B0

(C35) BFLOAT(%E);
(D35)                         2.7182818284590452353602874713526624977572470937B0

(D36)                                             BATCH DONE

(C37)
[DDT]
*
```

Conclusion

Even though MACSYMA can handle many problems and is generally enjoyable to use, it is not a cure-all. For example, it cannot solve all equations symbolically; some are simply too complicated for the system to handle. We have pointed out some of its other limitations throughout this technical document. However, the user can often write an algorithm using the MACSYMA language and available functions to solve particular problems numerically. One such case is our user-defined Newton-Raphson method that used the derivative function of MACSYMA to find zeros of a function.

In addition to the theoretical difficulties, which cause certain problems to be insolvable, practical problems are also involved. One of the most common of these problems is "intermediate expression swell." This often causes the user to run out of his allocated list space. With proper care and experience, however, the user can learn to avoid many of these problems.

It may seem that we have placed a great deal of emphasis on simplification and general manipulation of mathematical expressions; however, we have in reality only scratched the surface with regard to MACSYMA's capabilities in this area. This is, after all, the very heart of MACSYMA and must be thoroughly investigated by any serious user. It is, therefore, the hope of the authors that the reader will become interested enough to experiment with MACSYMA independently and discover the potentials of the system. As with any large programming system, MACSYMA's growth and development arises largely through user interaction.

References

1. *MACSYMA Reference Manual*, The Mathlab Group, Laboratory for Computer Science, Massachusetts Institute of Technology, Version Nine, Second Printing, December 1977.

2. Joel Moses, "Algebraic Simplification: A Guide for the Perplexed," *J. Comm. ACM* 14,8, August 1971, p. 527.

3. Churchill, Ruel J., *Complex Variables*, McGraw-Hill, 1960.

4. Forsythe, George and Cleve B. Moler, *Computer Solution of Linear Algebraic Systems*, Prentice-Hall, 1967.

Bibliography

Kaplan, Wilfred, *Advanced Calculus*, Addison-Wesley, 1952.

Lewis, Ellen, "An Introduction to ITS for the MACSYMA User," Mathlab Memo #3, Mathlab Group, Laboratory for Computer Science, Massachusetts Institute of Technology, January 9, 1978.

MacDuffee, Cyrus C., *Theory of Equations*, John Wiley & Sons, Inc., New York, 1954.

Proceedings of the 1977 MACSYMA Users' Conference held at the University of California, Berkeley, California, July 27-29, 1977, Scientific and Technical Information Office, National Aeronautics and Space Administration, Washington, DC, 1977.

Proceedings of the 1979 MACSYMA Users' Conference, Washington, DC, June 20-22, 1979, MIT Laboratory for Computer Science, Cambridge, MA, 1979.

Scheid, Francis, *Numerical Analysis*, Schaum's Outline Series, McGraw-Hill, 1968.

```
(C1) BATCH(DRI1,1,DSK,DRINK);

(C2) /*

                                   APPENDIX

          A PRACTICAL PROBLEM ILLUSTRATING MACSYMA'S UTILITY AS A LABOR SAVING TOOL

*/

DYNAMALLOC:TRUE;
(D2)                                          TRUE

(C3) /* Suppose that we want to find the minimum of the function: */

F(X,Y,Z,T):=X^2+2*Y^2+Z^2+T^2;
                                                   2       2    2    2
(D3)                              F(X, Y, Z, T) := X  + 2 Y  + Z  + T

(C4) /* subject to the conditions: */

X+3*Y-Z+T=2;
(D4)                                    - Z + 3 Y + X + T = 2

(C5) 2*X-Y+Z+2*T=4;
(D5)                                    Z - Y + 2 X + 2 T = 4

(C6) /* Solution:  Using the method of Lagrange multipliers, we form the function, */

G(X,Y,Z,T,LAMDA1,LAMDA2):=(X^2+2*Y^2+Z^2+T^2)+LAMDA1*(X+3*Y-Z+T-2)+LAMDA2*(2*X-Y+Z+2*T-4);
                                      2       2    2    2
(D6)   G(X, Y, Z, T, LAMDA1, LAMDA2) := X  + 2 Y  + Z  + T  + LAMDA1 (X + 3 Y - Z + T - 2) + LAMDA2 (2 X - Y + Z + 2 T - 4)

(C7) /* Next, we compute the 1st partial derivatives of the function G with respect to each independent variable, separately
setting each resulting expression equal to zero.  */

DIFF(G(X,Y,Z,T,LAMDA1,LAMDA2),X)=0;
(D7)                                  2 X + 2 LAMDA2 + LAMDA1 = 0

(C8) DIFF(G(X,Y,Z,T,LAMDA1,LAMDA2),Y)=0;
(D8)                                  4 Y - LAMDA2 + 3 LAMDA1 = 0

(C9) DIFF(G(X,Y,Z,T,LAMDA1,LAMDA2),Z)=0;
(D9)                                   2 Z + LAMDA2 - LAMDA1 = 0

(C10) DIFF(G(X,Y,Z,T,LAMDA1,LAMDA2),T)=0;
(D10)                                 2 T + 2 LAMDA2 + LAMDA1 = 0

(C11) DIFF(G(X,Y,Z,T,LAMDA1,LAMDA2),LAMDA1)=0;
(D11)                                  - Z + 3 Y + X + T - 2 = 0

(C12) DIFF(G(X,Y,Z,T,LAMDA1,LAMDA2),LAMDA2)=0;
(D12)                                  Z - Y + 2 X + 2 T - 4 = 0

(C13) /*
```

```
Solving these six linear equations in six unknowns is tedious but a routine step in problems of this type.  Here we see
a real labor saving tool in allowing MACSYMA to solve the system of equations for us.  */

SOLVE([%TH(6),%TH(5),%TH(4),%TH(3),%TH(2),%],[X,Y,Z,T,LAMDA1,LAMDA2]);

SOLVE FASL DSK MACSYM being loaded
Loading done
                                     67       2        14       67                 26                 18
(D13)                           [[X = --, Y = --, Z = --, T = --, LAMDA1 = - --, LAMDA2 = - --]]
                                     69       23       69       69                 69                 23

(C14) /* If we now evaluate the function F(X,Y,Z,T) at these solution values for the system of equations just solved, we obtain
the minimum of the function F(X,Y,Z,T) subject to the given conditions.  */

F(67/69,2/23,14/69,67/69);
                                                                134
(D14)                                                           ---
                                                                69

(C15) EV(%,NUMER);
(D15)                                                        1.94202898

(C16) /*  Since the purpose of this problem is simply to illustrate the labor saving utility of MACSYMA, the test to determine
whether or not we have obtained a true minimum value for the given function is left to the reader.  */

DYNAMALLOC:FALSE;
(D16)                                                          FALSE

(D17)                                                        BATCH DONE

(C18)
[DDT]
*
```

☆U.S. GOVERNMENT PRINTING OFFICE:1981-714-007/293

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER TD 6401	2. GOVT ACCESSION NO. AD-A098647	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MACSYMA: A PROGRAM FOR COMPUTER ALGEBRAIC MANIPULATION (DEMONSTRATIONS AND ANALYSIS)		5. TYPE OF REPORT & PERIOD COVERED 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) R. Drew Drinkard, Jr., and Nancy K. Sulinski		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Underwater Systems Center New London Laboratory New London, CT 06320		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 771Y00
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Underwater Systems Center Newport, RI 02840		12. REPORT DATE 10 March 1981 13. NUMBER OF PAGES 100
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

MACSYMA
Algebraic Manipulation
Symbolic Manipulation

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The MACSYMA algebraic manipulation system is a collection of programs embedded in a LISP interpreter. It is currently implemented on the Incompatible Timesharing System (ITS) that operates on a PDP-10 -- known as the MC (MACSYMA Consortium) machine. This program is currently available to all employees at the Naval Underwater Systems Center via the NAval Laboratory COmputer Network (NALCON) or the ARPANET.

This technical document provides the (potential) user of MACSYMA with a

20. (Continued)

collection of illustrative examples of MACSYMA commands and an analysis of their effectiveness from the point of view of a new user. This document is not intended to serve as an instruction manual on the use of MACSYMA, but as a supplement to such a manual for the purpose of highlighting some common problems associated with its use.

Initial Distribution List

ADDRESSEE	NO. OF COPIES
OUSDR&E (W. J. Perry)	1
Dep. USDR&E (Res. & Adv. Tech.) (R. M. Davis)	1
Dep. USDR&E (Dir. Elect. & Phys. Sc.) (L. Wiseberg)	1
OASN, Dep. Assit. Sec. (Dr. R. Hoglund)	1
ONR (ONR-200, -102, -212, -222, -230, -431)	6
CNO (OP-02, -03-EG, -090, -098, -224, -35, -902, -951, -96, -981F, -981G1, -981H, -981J, -982F)	14
CNM (MAT-08T2, -08T21, -08T24, SP-20, ASW-122)	5
DIA (DT-2C)	1
NAVSURFWEAPCTR, White Oak (Code R-44 (J. B. Bell, G. R. Shubin), Code K-64 (L. Hieb))	3
NAVSURFWEAPCTR, Dahlgren (K-21 (C. M. Blackmon), Code K-55 (R. Manrique), Code K-74 (A. H. Morris, H. Thombs)	4
NPRDC (Code P202 (L. Friedman))	1
DWTNSRDC, BETHESDA (Code 1805 (Dr. E. Cuthill (5)), Code 1826 (K. Meals (5)), Code 1844 (S. Dhir, P. Matula), Code 1892.1 (J. Strickland), Code 1892.2 (D. V. Sommer), Code 1619 (P. Montana), Code 19 (T. Eisler))	16
DWTNSRDC ANNA	1
DWTNSRDC CARD	1
NRL (Code 00, Code 1722 (G. Perez, L. Lin), Code 6020 (D. Book), Code 7926 (S. L. Coffey), Code 5308 (J. Coleman)	6
NRL, USRD	1
NRL, AESD	1
NAVAIRDEVCEN (Code 00, Code 2052, Code 60432 (C. Chen), Code 8511 (D. Tarrant))	4
NCSC (Code 732, D. Brown and Code 751, P. Bishop)	2
NORDA (Code 110 (Dr. R. Goodman), Code 301 (M. E. Gove))	1
USOC (Code 241, 240)	2
SUBASE LANT	1
NAVSUBSUPACNLON	1
NAVOCEANO (Code 02)	1
NAVELECSYSCOM (ELEX 00, 03, 310, 304)	4
NAVSEASYSCOM (SEA-003, -03R, -05R, -511, -06, -06D, -06R, -61, -61R, -62, -62R, -63, -63R, -63R-1, -63R-13, -63X, -631X, -631Y, -632X, -632Y, -902, -92R, -996)	23
NAVSEASYSDET, Norfolk	1
NOSC (Code 00, Code 6565 (Library), Code 8222 (W. Loper), Code 9122 (C. Messenger))	4
NAVWPNSCEN (Code 3132 (J. Zacharias), Code 31302 (Dr. J. Zenor), Code 62201 (R. Williams), Code 38 (R. Kamen))	4
CIVENGRLAB	1
NAVSURFWPNCEN	1
CHESNAVFACENGCOM (FPO-IP3)	1
NUWES, Keyport	1
NUWES, San Diego Detachment	1
NUWES, Hawaii Detachment	1

NAVAVIONICFAC	1
NAVTRAEQUIPCEN	1
NAVPGSCOL	1
NETC	1
NAVTRAEQUIPCENT (Tech. Library)	1
APL/UW, Seattle	1
ARL/PENN STATE, State College	1
ARL, University of Texas	1
DTIC	~~12~~ 2
DARPA	1
NOAA/ERL	1
National Research Council	1
Weapon System Evaluation Group	1
Woods Hole Oceanographic Institution	1
Engineering Societies Library, United Engineering Center	1
Marine Physical Lab, Scripps	1
MIT Laboratory for Computer Science, Math Lab (V. E. Golden)	3
RPI (Professor B. F. Caviness)	1
Aurora College (Professor H. Easley)	1
U.S. Army Research Office (Dr. P. T. Boggs)	1
ONR (R. L. Sternberg)	1
UCONN, Dept. of Math. U-9 (J. A. Roulier, M. Wachman)	2

LME